Formalisation et évaluation de politiques de sécurité et de respect de la vie privée en ELAN

MÉMOIRE

1er Juillet 2003

pour l'obtention du

DEA de l'Université Henri Poincaré – Nancy I

(Spécialité Informatique)

par

Vincent CRIDLIG

Composition du jury

Dominique Méry
Didier Galmiche
Noëlle Carbonell
André Schaff

Encadrant : Jacques Guyard

Laboratoire Lorrain de Recherche en Informatique et ses Applications — UMR 7503

Table des matières

Table des figures

Chapitre 1

Introduction

XML[1] (eXtensible Markup Language), ce langage qui permet aux applications de communiquer sur Internet via les web services, est devenu l'infrastructure favorite pour les applications de commerce électronique. La plupart de ces transactions entre les commerçants, les acheteurs et les fournisseurs nécessitent un mécanisme XML pour créer un climat de confiance et de sécurité entre ces différents partenaires. Mais, la sécurité liée au commerce en ligne ne se limite pas à un cryptage systématique ou à des systèmes d'authentification. De nombreux problèmes comme le respect de la vie privée viennent se greffer sur ces problèmes de sécurité. Les abus ne viennent pas seulement des pirates qui essaient de récupérer des données passant sur le réseau. Ils peuvent aussi venir des partenaires qui redistribuent parfois de façon abusive les données collectées grâce aux formulaires en ligne ou en analysant le comportement des visiteurs sur leur site web.

XML constitue un formidable outil pour apporter un meilleur niveau de confiance dans le commerce électronique principalement. De nombreux outils comme XKMS[2], WS-Security[3], pour les aspects d'identification et de chiffrement ou comme P3P[4] (Platform for Privacy Preferences) et APPEL[5] (A P3P Preference Exchange Language)[7] pour le respect de la vie privée fournissent une infrastructure de plus en plus complète pour gérer tous les aspects sécuritaires. En plus de tous ces modèles de sécurité applicatifs, on trouve des outils dans des couches plus basses qui assurent des services équivalents via des canaux sécurisés (IPSec, SSL, ...).

Le sujet consiste à étudier plusieurs aspects de la sécurité et plus particulièrement celui du respect de la vie privée. Pour exprimer les politiques qui concernent le traitement des données personnelles, le w3c a défini un langage (P3P) que nous avons formalisé en ELAN [11] pour construire ainsi un évaluateur. L'objectif du stage est de faire le tour des technologies qui gravitent autour de P3P et de développer un évaluateur ELAN pour avertir les internautes des pratiques des sites qu'ils visitent. Cet évaluateur ainsi que d'autres programmmes (un firewall [3] et une modèle réduit de Ponder [10]) doivent nous fournir la base pour apprécier la capacité d'ELAN à modéliser et évaluer les politiques de sécurité (et même les politiques réseau au sens général).

Le rapport est organisé comme suit : après avoir fait l'état de l'art sur toutes ces techniques dans le chapitre 2, l'intérêt s'est porté plus particulièrement sur l'utilisation d'un langage formel basé sur des règles de réécriture (ELAN). Le chapitre 3 présente les possibilités d'ELAN dans le cadre du protocole P3P ainsi que dans la réalisation d'un firewall. Le chapitre 4 concluera ce rapport en ouvrant quelques perspectives.

[1] `http://www.w3.org/TR/REC-xml`
[2] `http://www.w3.org/TR/xkms/`
[3] `http://www.ibm.com/developerworks/library/ws-secure/`
[4] `http://www.w3.org/TR/P3P/`
[5] `http://www.w3.org/TR/P3P-preferences/`

Chapitre 2

Etat de l'art : modèles de sécurité

2.1 Protection de la vie privée

Au cours des dernières années, les problèmes liés à la vie privée sur Internet sont devenus très importants aux yeux des utilisateurs. Par exemple, l'utilisation abusive des cookies dans les transactions Web reste un véritable problème [6]. De nombreux organismes internationaux se sont penchés sur la question, aussi bien du côté de la législation que du point de vue technologique [4]. Deux modèles génériques ont été proposés PICS[6] (Platform for Internet Content Selection) et P3P (Platform for Privacy Preference), ce dernier prenant largement le dessus aujourd'hui de par sa standardisation dans le W3C (World Wide Web Consortium)[8] [4].

Lorsqu'un utilisateur se connecte à un serveur web, il lui communique involontairement un certain nombre de données que ce soit sur son identité ou ses habitudes. Ces informations sont parfois recoupées avec celles d'autres sites pour établir un profil des visiteurs. Il en découle de la part des utilisateurs une forte attente en matière de transparence au niveau des pratiques des sites web. L'utilisateur serait plus tranquille s'il pouvait contrôler les informations qu'il fournit. En effet, il serait souhaitable de savoir quelles informations il accepte de communiquer, à qui, combien de temps ces informations sont conservées, à quoi elles servent... Cependant, on ne peut pas demander aux utilisateurs de s'informer sur les pratiques de chaque site qu'il visite. Ce serait évidemment trop pénible. De plus, ce phénomène est amplifié par les transactions cachées. En effet, une page HTML peut contenir plusieurs ressources venant de plusieurs sites donc plusieurs politiques potentielles.

P3P, qui est une recommandation du w3c, répond en partie à cette attente en donnant la possibilité aux sites web d'informer les utilisateurs de leurs politiques vis à vis du respect de la vie privée. Il définit un format standardisé pour décrire ces politiques. P3P fournit également un moyen à l'utilisateur pour localiser ces politiques et les comparer à ses préférences. Si les conditions de l'utilisateur sont respectées, celui-ci accède aux ressources. Sinon, il est averti grâce à une fenêtre pop-up et il a le choix de continuer ou non.

2.1.1 P3P

Généralités

P3P et la loi P3P a pour objectif de fournir un moyen standardisé aux sites web pour décrire leur politique. Ces politiques permettent d'annoncer l'utilisation des données collectées par ces sites webs. L'utilisateur, qui a défini ses préférences, peut alors télécharger de façon transparente

[6] `http://www.w3.org/PICS/`

la politique distante pour décider s'il l'accepte ou non. Mais il faut noter que P3P ne fait pas office de loi. Tout est basé sur la confiance que le visiteur accorde au site. P3P ne permet pas de savoir si un site respecte bien la politique annoncée. En cela, P3P privilégie l'auto-régulation sur Internet. Il n'y a pas d'entité supérieure qui contrôle ce respect effectif des politiques. L'auto-régulation vient du caractère international de l'Internet et du fait que les frontières sont difficilement définissables. Chaque pays ayant ses propres lois, il semble plus simple, pour l'instant, de définir une sorte de nouvel espace de lois qui s'auto-gère.

Rappel sur la nature des données en question Il est important de rappeler ici quel genre de données peuvent être collectées par un serveur et comment elles sont collectées. Il existe principalement trois types de données récupérables.

Tout d'abord, il est possible lors d'une connexion de récupérer plusieurs informations telles que l'adresse IP ou le nom de la machine, le système d'exploitation utilisateur, la version de son navigateur, ... Ces données sont contenues dans les en-têtes HTTP et sont accessibles par exemple en utilisant les variables PHP[7] (Hypertext Processor).

Ensuite, le serveur peut stocker dans un fichier le parcours suivi par le visiteur : quel fichier a-t-il consulté, en cliquant sur quelle image, quel lien... C'est ce qu'on appelle le clickstream. On utilise, pour faire cela, les logs du serveur qui stocke l'adresse IP du client, l'URI (Uniform Ressource Identifiers) de la ressource demandée, le temps de traitement, la méthode HTTP, ... En regroupant ces informations, on peut retrouver le chemin parcouru par le visiteur.

Enfin, les formulaires permettent de récupérer des données personnelles identifiantes ou non comme les nom, prénom, age, sexe, adresse, pays, métier, employeur, ... L'utilisateur donne ces informations volontairement ce qui diffère des deux premiers types de données.

En plus de tout cela, presque tous les sites déposent des cookies sur le disque dur des visiteurs encore une fois grâce aux en-têtes HTTP. Ceux-ci permettent de reconnaître un ancien visiteur et de permettre les sessions. Il existe plusieurs types de cookies :

- le cookie du site en cours (first-party cookie)
- les cookies des sites tiers (third-party cookies)

Un cookie peut être un cookie de session. Il ne dure que le temps de la visite. Il existe également les cookies persistants dont la durée de vie est fixée par le serveur. Rappelons qu'un cookie ne peut être vu que par le site qui l'a déposé. Un site web peut déposer plusieurs cookies, chaque cookie étant repéré par le domaine et le path (sous-arbre où le cookie est valide). À chaque requête d'une ressource incluse dans le path, le cookie est envoyé grâce à l'en-tête.

Rappel sur les utilisations possibles de ces données Le premier type de données vu précédemment (version du navigateur, système d'exploitation, ...) permet d'optimiser les pages pour le système et le navigateur utilisateur. Le deuxième type (données dynamiques, clicks utilisateur, chemin parcouru) permet quant à lui de faire des statistiques sur les pages consultées, le chemin suivi par l'utilisateur, les liens les plus souvent cliqués. Tout cela dans le but d'améliorer le design du site et de personnaliser le site pour chaque utilisateur connaissant ces préférences. Le troisième type d'information permet de mieux identifier le client pour faire par exemple des transactions ou encore des statistiques (âge moyen des visiteurs, ...). Les cookies de session permettent de stocker l'identifiant de l'utilisateur pour lui accorder un droit d'accès dans un espace sécurisé par exemple. Les cookies persistants servent à retrouver le profil de l'utilisateur grâce à l'identifiant unique stocké dans le cookie. Ainsi, le serveur retrouve les données concernant ce visiteur et peut personnaliser les pages.

[7] `http://www.php.net`

Toutes ces utilisations ne sont que des exemples ; il existe beaucoup d'autres utilisations possibles.

Qu'est-ce que P3P ? P3P est une recommandation du w3c visant à formaliser en langage XML les pratiques des sites webs. Avant P3P, les sites se contentaient de décrire leurs pratiques en langage naturel (cf. Annexes). P3P permet de convertir ces pratiques en XML de façon à automatiser la comparaison entre les préférences des visiteurs et les pratiques des sites web. P3P définit également des moyens pour l'utilisateur de localiser ces politiques dans l'arborescence d'un site.

P3P à travers un exemple

Déploiement côté serveur Mettons-nous à la place d'un site web qui voudrait déployer P3P pour acquérir la confiance de ses visiteurs. Ce site a déjà décrit ses pratiques en langage naturel. Il faut donc faire la conversion de ces pratiques en respectant la syntaxe P3P. De plus, il faudra créer un fichier de références de politiques pour permettre aux agents utilisateurs de savoir quelles politiques s'appliquent à quelles parties du site. En effet, il est possible de définir plusieurs politiques.

Notre site s'appelle librairie.com et vend toutes sortes de livres, magazines et journaux en ligne. Le site dispose d'un catalogue consultable librement, d'un espace sécurisé pour l'identification de l'acheteur, l'achat et la livraison. Le site utilise également les cookies pour adapter les publicités aux goûts du visiteur.

Première politique On va définir une première politique pour la partie sécurisée du site.

- Cette politique est déjà décrite en langage naturel à l'URL `http://www.librairie.com/privacy/policy.html` :
  `discuri = "http://www.librairie.com/privacy/policy.html"`
- On nomme cette politique policy1 : name="policy1"
- Elle est valable pendant 10 jours. Donc si un utilisateur revient sur le site après 15 jours, il devra recharger la politique :
  `<EXPIRY max-age="864000"/>`
- Cette société est basée à Paris, 14 rue du net et on peut la contacter soit par telephone au 0112123456 ou par mail à l'adresse info-contact@libraire.com :

```
<DATA-GROUP>
 <DATA ref="#business.name">Librairie</DATA>
 <DATA ref="#business.department">75</DATA>
 <DATA ref="#business.contact-info.postal.street">rue du net</DATA>
 <DATA ref="#business.contact-info.postal.city">Paris</DATA>
 <DATA ref="#business.contact-info.postal.stateprov">France</DATA>
 <DATA ref="#business.contact-info.postal.postalcode">75000</DATA>
 <DATA ref="#business.contact-info.telecom.telephone.number">0112123456</DATA>
 <DATA ref="#business.contact-info.online.email">info-contact@libraire.com</DATA>
</DATA-GROUP>
```

- Le site permet aux utilisateurs d'accéder à n'importe quel moment à leurs données personnelles :
  `<ACCESS><all/></ACCESS>`
- En cas de litige, les utilisateurs peuvent contacter le service clientèle à l'URL `http://juridique.librairie.com`. De plus, les problèmes seront résolus par le service :
  `<DISPUTES-GROUP>`

```
<DISPUTES resolution-type="service"
          service="http://juridique.libraire.com">
  <IMG src="http://image.jpg"/>
  <REMEDIES><correct/></REMEDIES>
</DISPUTES>
</DISPUTES-GROUP>
```

- Si le visiteur décide d'acheter un produit,
  ```
  <CONSEQUENCE>
      Nous utilisons ces informations quand vous achetez un produit.
  </CONSEQUENCE>
  ```
- l'information est collectée pour l'activité demandée uniquement :
  `<PURPOSE><current/></PURPOSE>`
- le site ne communique pas ces informations à des tiers excepté à la société de livraison mais celle-ci n'utilise pas ces informations dans d'autres buts que la livraison courante :
  `<RECIPIENT><ours/></RECIPIENT>`
- le site ne garde ces données que le temps de la transaction :
  `<RETENTION><stated-purpose/></RETENTION>`
- on lui demande son nom, prénom, adresse, téléphone, email, login et mot de passe pour pouvoir accéder à ses données personnelles, le numéro de sa carte de crédit :
  ```
  <DATA-GROUP>
              <DATA ref="#user.name"/>
              <DATA ref="#user.home-info.postal"/>
              <DATA ref="#user.home-info.telecom.telephone"/>
              <DATA ref="#user.business-info.postal"/>
              <DATA ref="#user.business-info.telecom.telephone"/>
              <DATA ref="#user.home-info.online.email"/>
              <DATA ref="#user.login.id"/>
              <DATA ref="#user.login.password"/>
              <DATA ref="#dynamic.miscdata">
               <CATEGORIES><purchase/></CATEGORIES>
              </DATA>
  </DATA-GROUP>
  ```

Si on résume, voici les balises nécessaires à l'écriture d'une politique (certaines sont optionnelles) :

- EXPIRY : date d'expiration ou durée de validité
- ENTITY : description de l'entité légale qui expose ses pratiques
- ACCESS : indique si le site offre des possibilités d'accès aux données
- DISPUTES-GROUP : description des procédures de résolution de conflit en cas de réclamation concernant les pratiques du site
- REMEDIES : description des réparations possibles en cas de violation de la politique
- STATEMENT :
  - CONSEQUENCE : explication en langage naturel
  - PURPOSE : but de la collecte de ces données (ex : administration du site, recherche et développement, contact, accomplissement d'une transaction, ...)
  - RECIPIENT : destinataires des données collectées (ex : le site et ses agents, service de livraison, entités tierces suivant les mêmes pratiques, ...)
  - RETENTION : durée de conservation des données (ex : données conservées juste le temps de la session, le temps de finir l'action demandée, indéfiniment, ...)

– DATA-GROUP : type de données collectées

Deuxième politique Pour la partie catalogue, le site utilise les données fournies par le navigateur du visiteur pour adapter l'affichage des pages. Il collecte également des informations concernant le parcours du visiteur, le temps de consultation des pages, les images cliquées, tout ceci pour améliorer le site. Et pour le personnaliser lors des prochaines visites de l'utilisateur, il utilise des cookies pour répondre aux préférences de l'utilisateur.

```
<POLICY discuri = "http://www.librairie.com/privacy/policy.html"
     name="policy2">

  <EXPIRY max-age="864000"/> <!-- 10 days -->
  <ENTITY>
    <DATA-GROUP>
      <DATA ref="#business.name">Librairie</DATA>
      <DATA ref="#business.department">75</DATA>
      <DATA ref="#business.contact-info.postal.street">rue du net</DATA>
      <DATA ref="#business.contact-info.postal.city">Paris</DATA>
      <DATA ref="#business.contact-info.postal.stateprov">France</DATA>
      <DATA ref="#business.contact-info.postal.postalcode">75000</DATA>
      <DATA ref="#business.contact-info.telecom.telephone.number">0112123456</DATA>
      <DATA ref="#business.contact-info.online.email">info-contact@libraire.com</DATA>
    </DATA-GROUP>
  </ENTITY>
  <ACCESS><non-ident/></ACCESS>
  <DISPUTES-GROUP>
    <DISPUTES resolution-type="service" service="http://juridique.libraire.com">
      <IMG src="http://image.jpg"/>
      <REMEDIES><correct/></REMEDIES>
    </DISPUTES>
  </DISPUTES-GROUP>
  <STATEMENT>
    <PURPOSE><current/><admin/><develop/></PURPOSE>
    <RECIPIENT><ours/></RECIPIENT>
    <RETENTION><indefinitely/><RETENTION>
    <DATA-GROUP>
      <DATA ref="#dynamic.clickstream"/>
      <DATA ref="#dynamic.http"/>
    </DATA-GROUP>
  </STATEMENT>
  <STATEMENT>
    <CONSEQUENCE>
       Nous personnalisons le site en fonction de vos précédentes visites.
    </CONSEQUENCE>
    <PURPOSE><tailoring/><develop/></PURPOSE>
    <RECIPIENT><ours/></RECIPIENT>
    <RETENTION><stated-purpose/></RETENTION>
    <DATA-GROUP>
      <DATA ref="#dynamic.cookies">
        <CATEGORIES><state/></CATEGORIES>
      </DATA>
      <DATA ref="#dynamic.miscdata">
        <CATEGORIES><preference/></CATEGORIES>
      </DATA>
```

```
    </DATA-GROUP>
   </STATEMENT>
</POLICY>
```

Les données de type #dynamic.cookies et #dynamic.miscata sont "unstructured" donc il faut préciser leur catégorie. Ces données font partie des structures de données de base fournies par P3P. Il est possible d'en définir d'autres avec la balise DATASCHEMA soit dans le fichier contenant les politiques, soit dans un fichier à part. DATA-STRUCT permet de définir la structure arborescente abstraite des données, par exemple :

```
<DATASCHEMA xmlns="http://www.w3.org/2002/01/P3Pv1">
   <DATA-STRUCT name="document.auteur"
       short-description="Auteur"
       <CATEGORIES><preference/></CATEGORIES>
   </DATA-STRUCT>
   <DATA-STRUCT name="document.dateParution"
       short-description="Date de parution"
       <CATEGORIES><preference/></CATEGORIES>
   </DATA-STRUCT>
   ...
   <DATA-DEF name="livre" structref="#document"/>
   <DATA-DEF name="journal" structref="#document"/>
</DATASCHEMA>
```

La CATEGORY préference indique que ce sont des données qui concernent les goûts de l'utilisateur. Si on avait défini user.telephone, on aurait utilisé la CATEGORY physical... Ce sont les mêmes CATEGORIES que celles de la spécification APPEL, ce qui permet de comparer les deux documents (politique du site et préférences utilisateur)

Il reste maintenant à créer le fichier des références de politiques qui permettra à l'utilisateur de trouver ces politiques :

```
<META xmlns="http://www.w3.org/2000/12/P3Pv1">
  <POLICY-REFERENCES>
    <EXPIRY max-age="864000"/> <!-- 10 days -->
    <POLICY-REF about="/P3P/Policies.xml#policy1">
      <INCLUDE>/achat/*</INCLUDE>
    </POLICY-REF>
    <POLICY-REF about="/P3P/Policies.xml#policy2">
      <INCLUDE>/*</INCLUDE>
      <EXCLUDE>/achat/*</EXCLUDE>
      <COOKIE-INCLUDE name="*" value="*" domain="*" path="*"/>
    </POLICY-REF>
  </POLICY-REFERENCES>
```

Ce fichier indique que toutes les ressources du sous-arbres /achat/* sont couvertes par la politique policy1. Et toutes les ressources sauf celles du sous-arbres /achat/* sont couvertes par la politique policy2. Tous les cookies sont également couverts par la deuxième politique.

Pour que l'utilisateur puisse accéder au site, il reste à placer ce fichier de références des politiques à un endroit prédéfini : /w3c/p3p.xml. On peut également ajouter un en-tête p3p : policy-ref-field à toutes les réponses du serveur avec comme valeur l'URL de ce fichier. Troisième possibilité : ajouter une balise link avec comme attributs href='URI du fichier des références de politiques' et rel='P3Pv1' dans toutes les pages HTML.

Les politiques compactes Les politiques compactes sont les résumés des politiques définies précédemment et permettent d'optimiser les performances. Mais, ceci est optionnel et il est toujours possible pour un agent utilisateur de récupérer la politique complète en cas de doute. Les politiques compactes ne concernent que les cookies et sont incluses dans les en-têtes HTTP. Chaque fois qu'un site dépose un cookie, l'en-tête *P3P : compact-policy-field* est envoyé avec comme valeur la politique compacte.

L'idée est de concaténer tous les éléments pour réduire la taille de la politique :

`<ACCESS>, <CATEGORIES>, <DISPUTES>, <NON-INDENTIFIABLE>,`
`<PURPOSE>, <RECIPIENT>, <REMEDIES>, <RETENTION>`

Si on reprend la deuxième politique de l'exemple précédent, on aura :
"NON DSP DEV TAI OUR STP STA PRE"
Voyons à quoi correspondent ces tokens :
- NON : non-ident pour la balise ACCESS
- DSP : présence de la balise DISPUTE-GROUP
- DEV TAI : tailoring et develop dans la balise PURPOSE
- OUR : ours dans la balise RECIPIENT
- STP : stated-purpose dans la balise RETENTION
- STA PRE : state et preference pour la catégorie

2.1.2 APPEL

APPEL (A P3P Preference Exchange Language) est un langage proche de P3P défini par le w3c. Il vise à fournir un moyen aux internautes de décrire leurs préférences personnelles en matière d'utilisation de leurs données. Ces préférences sont décrites en langage XML. Evidemment, on ne peut pas demander à un utilisateur basique d'écrire un fichier XML contenant des connecteurs logiques, des expressions régulières et des balises et attributs bien définis. Donc, il est possible d'importer des préférences par défaut répondant aux attentes de la plupart des utilisateurs. Mais rien n'empêche un utilisateur de définir son propre fichier APPEL.

Le fichier de préférences APPEL contient un ensemble de règles (balise RULE) regroupées dans une même balise RULESET. Une règle est caractérisée par un comportement (attribut behavior) à adopter en cas de succès. Le w3c propose 3 comportements :
- request : la politique du site est acceptable
- limited : l'accès à la ressource devrait être limité
- block : l'accès à la ressource ne devrait pas être autorisé

Voici un extrait d'un fichier APPEL qui indique que l'utilisateur accepte les cookies qui sont déposés dans le but d'adapter le contenu de la page (tailoring) et dont le destinataire sera uniquement le site lui-même (ours). La catégorie state signifie que les cookies permettent de gérer des états dans un protocole qui est sans état (http) :

```
<appel :RULE behavior="request">                                          1
      <p3p :POLICY>                                                       2
            <p3p :STATEMENT>                                              3
                  <p3p :RECIPIENT appel :connective="and">                4
                        <p3p :ours/>                                      5
                  </p3p :RECIPIENT>                                       6
                  <p3p :PURPOSE appel :connective="non-and">              7
                        <p3p :tailoring/>                                 8
                  </p3p :PURPOSE>                                         9
                  <p3p :DATA-GROUP>                                       10
```

```
                        <p3p :DATA ref="#dynamic.cookies">                    11
                              <p3p :CATEGORIES appel :connective="or">        12
                                    <state/>                                  13
                              </p3p :CATEGORIES>                              14
                        </p3p :DATA>                                          15
                  </p3p :DATA-GROUP>                                          16
            </p3p :STATEMENT>                                                 17
      </p3p :POLICY>                                                          18
</appel :RULE>                                                                19
```

Cet exemple montre que le langage APPEL utilise les mêmes balises que celles utilisées dans le langage P3P. La structure du fichier des préférences utilisateur est très proche de celle d'une politique P3P, ce qui facilite la comparaison de ces deux fichiers.

2.1.3 Interaction serveur/utilisateur

Supposons que le webmaster a configuré son site pour que chaque réponse contienne un en-tête *p3p : policy-ref-field*. L'interaction entre l'utilisateur et le serveur se déroule comme suit :

- Un visiteur demande une page web à un serveur.
- Le serveur envoie la page en précisant dans l'en-tête *P3P : policy-ref-field* l'URI du *policy reference file*.
- L'agent utilisateur demande le policy reference file s'il n'en a pas un de valide.
- Le serveur envoie la page demandée.
- L'agent utilisateur parse le *policy reference file* pour trouver la politique qui s'applique à la page initialement demandée et la demande au serveur si elle n'est pas incluse dans le *policy reference file*.
- Le serveur envoie la politique demandée.
- L'agent utilisateur parse la politique pour la comparer aux préférences de l'utilisateur. Si elle est conforme aux préférences, il affiche la page initiale sinon il peut avertir l'utilisateur avec une fenêtre pop-up par exemple

Si le site n'est pas configuré pour ajouter les en-têtes P3P à chaque réponse mais que celui-ci utilise le *well-known location*, l'agent utilisateur demande le *policy reference file*, puis la politique voulue avant d'afficher éventuellement la page. Enfin, si le site utilise la balise link ajoutée dans chaque fichier, le déroulement est le même que précédemment.

La figure 2.1 montre l'interaction entre l'agent utilisateur et le serveur web.

Notons que l'évaluation est asynchrone : cela signifie que pendant le temps de navigation où la politique n'a pas encore été chargée et évaluée, l'agent utilisateur considère que le site n'implante pas P3P. Ensuite, le processus d'avertissement de l'utilisateur est déclenché. Cependant, pour limiter les abus, le w3c recommande de placer les fichiers P3P dans une zone sûre qui ne dépose pas de cookies et qui collecte un minimum de données.

2.1.4 Les outils existants pour P3P

Il existe aujourd'hui plusieurs outils relatifs aux langages P3P et APPEL (voir le site du W3C pour la liste complète). La plupart des outils disponibles sont des éditeurs de règles. Ceux-ci créent une politique, en posant aux webmasters des questions sur l'utilisation des données. Ceci permet de générer rapidement une politique P3P en évitant le facteur de difficulté lié à la rédaction d'un tel document. Le risque est que le document généré ne soit pas suffisament précis

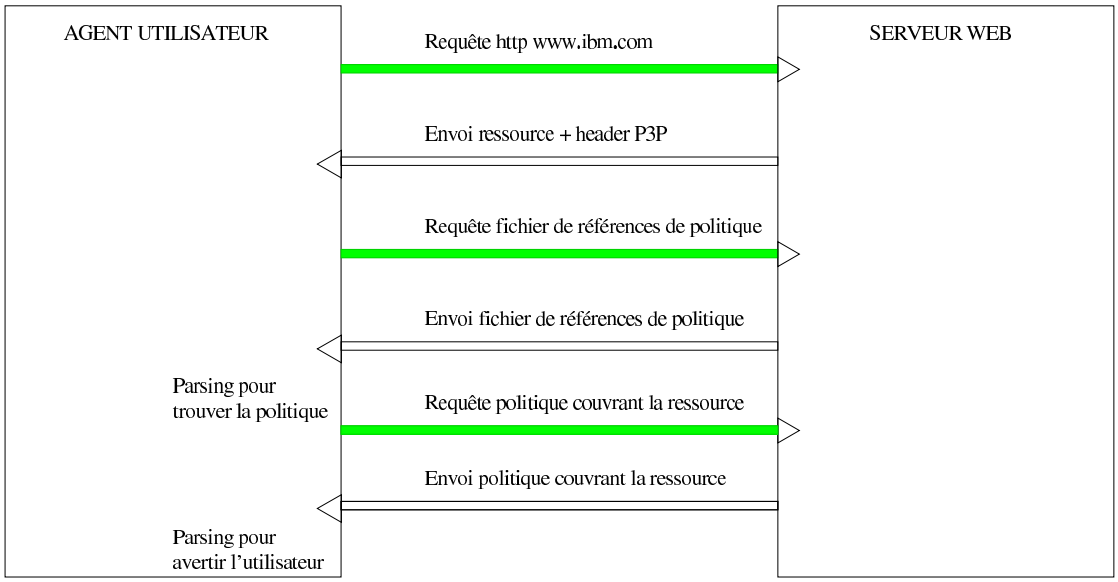

FIG. 2.1 – Interaction agent utilisateur / Serveur web

et en accord avec la politique réelle du site web. Comme exemple d'éditeur de politique P3P, on peut citer l'éditeur d'IBM p3peditor[8], p3pwriter[9] et p3pedit[10].

Quelques évaluateurs sont également intégrés dans les navigateurs Netscape 7.0 et Internet Explorer 6.0. Cependant, ces évaluateurs ne gèrent que les politiques compactes. Il existe aussi le plug-in gratuit (Privacy Bird) d'AT&T qui gère lui les vraies politiques P3P. Ce plug-in ne bloque jamais l'accès à une ressource car il a été pensé comme un outil d'avertissement. Il informe l'utilisateur d'une non-conformité en affichant un oiseau dont la couleur varie en fonction du résultat de l'évaluation et en jouant éventuellement un son d'oiseau adapté.

2.2 Sécurité applicative basée sur XML

2.2.1 XML Signature

La signature électronique est devenue incontournable dans les modèles de sécurité actuels. Elle conjugue plusieurs mécanismes et permet ainsi de répondre à différents critères de sécurité dont la confidentialité et l'intégrité.

Rappels

Les mécanismes utilisés par la signature électronique sont les algorithmes asymétriques (rsa, ..) et les fonctions de hachage (md5, sha, ...).

Un algorithme asymétrique utilisé entre deux entités nécessite deux jeux de clés. Chaque entité possède deux clés liées : une clé privée et une clé publique. Tout document chiffré avec l'une de ces deux clés peut être déchiffré avec l'autre clé. Ainsi, un document chiffré avec une clé privée peut être déchiffré par tout le monde, mais cela garantit l'authenticité du document. Inversement, un document chiffré avec la clé publique d'une autre entité ne pourra être déchiffré

[8] `http://www.alphaworks.ibm.com/tech/p3peditor`
[9] `http://www.p3pwriter.com/WzStp1.asp`
[10] `http://p3pedit.com/`

que par le possesseur de cette clé publique car il est le seul à posséder la clé privée associée. Ce mécanisme garantit la confidentialité.

Une signature électronique est associée à un document. Pour créer cette signature, on utilise une fonction mathématique de hachage qui calcule l'empreinte d'un document. La moindre modification du document entraîne le changement de l'empreinte. Il n'est pas possible de recréer le (ou les) document(s) à partir d'une empreinte donnée. L'empreinte permet au destinataire de vérifier l'intégrité du document. Si l'on chiffre maintenant cette empreinte avec notre clé privée, on assure l'intégrité et l'authentification. En effet, le destinataire pourra à son tour calculer l'empreinte du document original et la comparer avec l'empreinte obtenue en déchiffrant la signature avec la clé publique de l'émetteur.

Présentation de XML Signature

Dans une signature XML[11][9], on retrouve l'ensemble des informations nécessaires à la génération d'une signature, l'empreinte, la signature elle-même et d'autres algorithmes spécifiques à XML comme la canonisation[12]. Voyons plus en détail l'exemple fourni dans la spécification.

```
<Signature Id="MyFirstSignature" xmlns="http://www.w3.org/2000/09/xmldsig#">

  <SignedInfo>
    <CanonicalizationMethod
        Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-20010315"/>
    <SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#dsa-sha1"/>
    <Reference URI="http://www.w3.org/TR/2000/REC-xhtml1-20000126/">
      <Transforms>
        <Transform Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-20010315"/>
      </Transforms>
      <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
      <DigestValue>j6lwx3rvEPO0vKtMup4NbeVu8nk=</DigestValue>
    </Reference>
  </SignedInfo>

  <SignatureValue>MC0CFFrVLtRlk=...</SignatureValue>

  <KeyInfo>
    <KeyValue>
      <DSAKeyValue>
        <P>...</P><Q>...</Q><G>...</G><Y>...</Y>
      </DSAKeyValue>
    </KeyValue>
  </KeyInfo>

</Signature>
```

Une signature XML est composée de trois blocs.

Le premier bloc contient les méthodes de calcul de la signature. Dans cet exemple, on utilise une méthode de canonisation qui permet de formater un document XML (c'est-à dire mettre un document XML dans sa forme canonique). C'est important car le moindre différence entre deux documents XML qui représentent les mêmes données peut faire varier l'empreinte. Les applications qui manipulent des fichiers XML modifient parfois ceux-ci très légèrement (notamment au

[11] http://www.w3.org/TR/xmldsig-core/
[12] http://www.w3.org/TR/2001/REC-xml-c14n-20010315

niveau des blancs, des sauts de lignes, du codage, des éléments vides qu'il faut convertir en balise ouvrante et fermante, déclarations d'espace de nommage superflus, ...). La canonisation est le mécanisme qui permet de rendre physiquement identiques deux documents qui le sont logiquement. XML autorisant de nombreux changements syntaxiques (blancs, espaces de noms, éléments vides, attributs par défaut, ...), un même document peut avoir plusieurs formes. Ce phénomène complique la comparaison de deux documents XML logiquement équivalent et fait varier l'empreinte. La canonisation limite le problème car elle permet d'obtenir la même empreinte pour deux documents équivalents. Voici un exemple de canonisation. Le document à canoniser :

```
<?xml version="1.0"?>

<?xml-stylesheet   href="doc.xsl"
   type="text/xsl"   ?>

<!DOCTYPE doc SYSTEM "doc.dtd">

<doc>Hello, world!<!-- Comment 1 --></doc>

<?pi-without-data     ?>

<!-- Comment 2 -->

<!-- Comment 3 -->
```

Le même document canonisé :

```
<?xml-stylesheet href="doc.xsl"
   type="text/xsl"   ?>
<doc>Hello, world!</doc>
<?pi-without-data?>
```

On définit ensuite la méthode utilisée pour la signature. Ici, on se sert de DSA avec comme algorithme de calcul d'empreinte : sha. Puis, on indique l'URI du document que l'on va signer. Ce document va subir plusieurs transformations pour créer l'empreinte. Ici, on a simplement une canonisation puis le calcul de l'empreinte juste après. Cette empreinte est contenue dans l'élément DigestValue.

Le second bloc contient la valeur de la signature obtenue après chiffrement de l'empreinte avec la clé privée.

Le troisième bloc contient des informations relatives à la clé utilisée. On peut notamment trouver la valeur d'une clé ou encore un moyen de vérifier que l'utilisateur est bien possesseur de cette clé (identifiants de certificats X509).

2.2.2 XML Encryption

XML Encryption[13] est une recommandation du w3c qui permet de crypter des portions de document XML. Lorsqu'un fichier contient des données sensibles telles que un login et un mot de passe, il est souhaitable de crypter ces données. Prenons l'exemple suivant qui fournit des informations sur une personne : nom, prénom et identifiant (login + passwd) :

```
<?xml version='1.0'?>
```

[13] http://www.w3.org/TR/xmlenc-core/

```
<Personne xmlns='http://example.org/identification'>
  <Nom>Cridlig</Nom>
  <Prenom>Vincent</Prenom>
  <Identifiant>
    <Login>cridligv</Login>
    <Passwd>monMotDePasse</Passwd>
  </Identifiant>
</Personne>
```

Il est possible de crypter par exemple le contenu de l'élément Identifiant :

```
<?xml version='1.0'?>
<Personne xmlns='http://example.org/identification'>
  <Nom>Cridlig</Nom>
  <Prenom>Vincent</Prenom>
  <Identifiant>
    <EncryptedData Type='http://www.w3.org/2001/04/xmlenc#Element'
       xmlns='http://www.w3.org/2001/04/xmlenc#'>
      <CipherData>
        <CipherValue>A23B45C56</CipherValue>
      </CipherData>
    </EncryptedData>
</Identifiant>
</Personne>
```

On peut également utiliser un algorithme utilisant une clé elle-même cryptée dans ce document. Pour la décrypter, on a recours à l'élément keyInfo qui contient par exemple le nom de la clé à utiliser.

De même, on peut faire du double cryptage en cryptant une portion de document contenant déjà des données cryptées. Cependant, on ne peut pas crypter une partie des éléments définissant déjà une donnée cryptée afin de respecter la DTD. Par exemple, CypherData seul ne peut être crypté.

2.2.3 XKMS

XKMS (XML Key Management Specification) [14] est composé de deux protocoles : X-KISS et X-KRSS. XKMS permet de faciliter la gestion des clés pour le client.

X-KISS

X-KISS est un protocole qui permet de sous-traiter le processus de résolution de clé à partir des informations venant d'une signature ou d'un cryptage XML. Supposons qu'un client ait besoin de récupérer la clé publique correspondant à un document signé et que la signature XML contienne un élément RetrievalMethod qui correspond à un certificat X509. Il demande alors au serveur de confiance donné par l'attribut URI de lui envoyer la clé publique. Le serveur est capable de récupérer le certificat de façon transparente pour le client. Une fois le certificat récupéré, le serveur peut en extraire la clé publique et la renvoyer au client. X-KISS définit la façon de construire les requêtes à envoyer à un serveur. Dans le cas précédent, on aurait :

```
<Locate>
  <Query>
```

```
    <ds:KeyInfo>
      <ds:RetrievalMethod
         URI="http://www.PKeyDir.test/Certificates/01293122"
         Type="http://www.w3.org/2000/09/xmldsig#X509Data"/>
    </ds:KeyInfo>
  </Query>
  <Respond>
    <string>KeyName</string>
    <string>KeyValue</string>
  </Respond>
</Locate>
```

Dans cet exemple, le client a besoin du nom de la clé ainsi que sa valeur. Pour cela, X-KISS définit la balise Respond qui indique ce que doit contenir la réponse du serveur.

X-KRSS

X-KRSS est un protocole qui permet à des clients de s'inscrire (register) à un service de gestion de clé. X-KRSS communique avec un PKI (Public Key Infrastructure) qui gère les clés publiques et les lie aux informations du possesseur de cette clé. Les clés (publique et privée) peuvent être générées soit par le client soit par le serveur. Si le client génère les clés, il s'inscrit en fournissant sa clé publique et d'autres informations pour éventuellement se désinscrire plus tard. De plus, il signe ce message en utilisant sa clé privée pour prouver qu'il est bien le possesseur de cette paire de clés. Si c'est le serveur qui génère les clés, le client s'inscrit en demandant en retour la paire de clés :

```
<Register>
  <Prototype Id="keybinding">
    <Status>Valid</Status>
    <KeyID>mailto:Alice@cryptographer.test</KeyID>
    <KeyInfo>
      <ds:KeyInfo>
  <ds:KeyName>mailto:Alice@cryptographer.test</ds:KeyName>
      </ds:KeyInfo>
    </KeyInfo>
    <PassPhrase>Pass</PassPhrase>
  </Prototype>
  <AuthInfo>
    <AuthServerInfo>
      <KeyBindingAuth>
        <ds:Signature  URI="#keybinding" [HMAC-SHA1 (Prototype, Auth)] />
      </KeyBindingAuth>
    </AuthServerInfo>
  </AuthInfo>
  <Respond>
     <string>KeyName</string>
     <string>KeyValue</string>
     <string>Private</string>
  </Respond>
```

```
</Register>
```

X-KRSS fournit également un moyen pour se désinscrire et pour retrouver une clé privée perdue par le client.

XKMS facilite ainsi le développement des applications utilisant XML Signature et XML Encryption en fournissant un moyen simple pour retrouver les informations relatives à ces mécanismes.

2.2.4 WS-Security

WS-Security (Web Services Security)[1], proposé par IBM, décrit des extensions SOAP [13] pour construire des web services sécurisés. WS-Security permet l'envoi de données sensibles (nom, identifiant, clé, ...), l'intégrité et la confidentialité des messages. Cette spécification définit quelques nouvelles balises pour intégrer l'intégrité (en utilisant XML Signature) et la confidentialité (en utilisant XML Encryption) dans des messages SOAP. WS-Security utilise l'en-tête des messages SOAP pour stocker par exemple :

- la signature d'une partie du corps de l'enveloppe,
- la liste des éléments chiffrés du message identifiés par l'attribut ID,
- un certificat X509,
- des Security Tokens qui permettent d'annoncer des informations sur l'utilisateur.

WS-Security finalise en quelque sorte toutes les spécifications relatives à la sécurité basées sur XML, en spécifiant la façon d'intégrer XML Signature et XML Encryption dans des messages SOAP. Pour résumer, WS-Security repose sur XML Signature, XML Encryption et SOAP. XML Signature repose lui sur des mécanismes comme les certificats, les fonctions de hachage et les algorithmes à clé publique. XML Encryption est basé sur des algorithmes de chiffrement. La figure 2.2 schématise la hiérarchie de tous ces mécanismes. XKMS n'y apparaît pas car il n'est pas vraiment sur le même plan. Il facilite simplement la recherche d'informations qui concernent les clés.

2.3 Sécurité aux niveaux inférieurs

2.3.1 SSL (Secure Socket Layer)

La couche SSL se trouve entre les couches TCP (Transport Control Protocol) et des couches applicatives comme HTTP (HyperText Transport Protocol), LDAP (Lightweight Directory Access Protocol), POP (Post Office Protocol), IMAP (Internet Message Access Protocol). SSL permet de chiffrer les données sensibles (mots de passe, formulaires postés par les clients, mails, ...), permet d'authentifier les entités en présence et garantit l'intégrité des messages. Pour répondre à ces besoins, SSL utilise les mêmes mécanismes que ceux vus précédemment pour WS-Security :

- confidentialité : utilisation d'une clé secrète partagée obtenue après la phase de négociation,
- authentification : utilisation des algorithmes à clé publique,
- intégrité : calcul d'une empreinte du message chiffrée avec la clé privée de l'émetteur.

SSL est divisé en deux sous-couches. SSL handshake protocol réalise la phase d'authentification (échange des certificats X509 et preuve de possession) et de négociation sur le choix des algorithmes. SSL record protocol encapsule les paquets des applications qui utilisent SSL.

TLS est l'équivalent de SSL mais il est proposé par l'IETF alors que SSL vient de Netscape. Les deux protocoles sont très proches et compatibles.

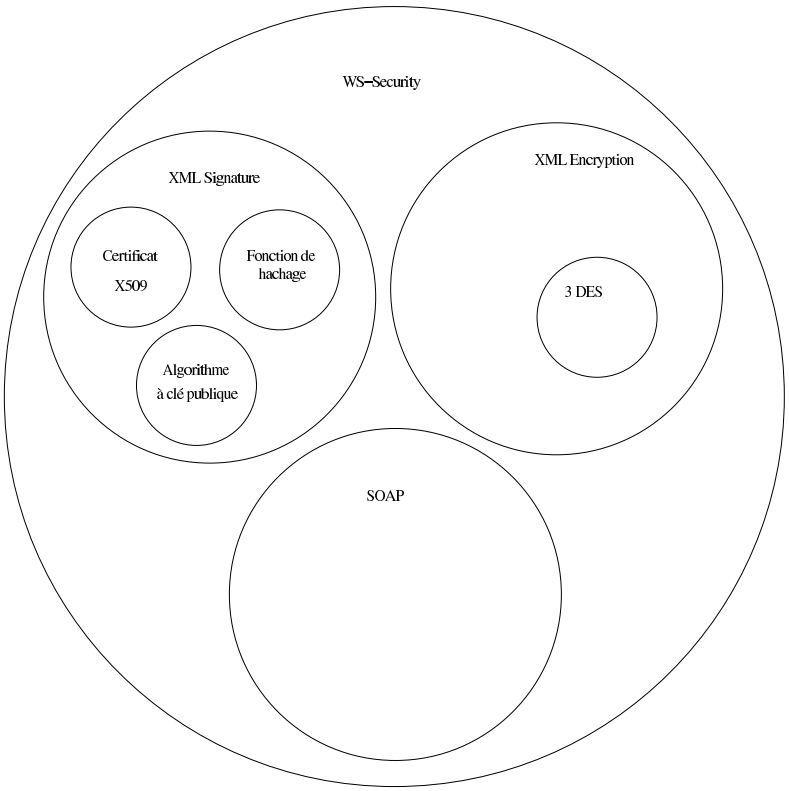

FIG. 2.2 – Récapitulatif du fonctionnement de WS-Security

2.3.2 IPSec

Si l'on descend encore d'un niveau dans la pile TCP/IP, on a un autre apport pour la sécurité qui est IPSec. Contrairement à tous les protocoles précédents dits de bout en bout, IPSec fonctionne au coeur du réseau. Il est principalement utilisé dans le cadre des réseaux privés virtuels (VPN) qui consistent à encapsuler les paquets IP. Ce processus est souvent appelé tunneling. Les propriétés de sécurité réalisées par IPSec sont :

- Confidentialité et intégrité des données,
- Authentification des entités aux extrémités,
- Chiffrement des adresses IP,
- Anti-rejeux, protection contre les attaques de type man-in-the-middle.

IPSec met en place deux types de tunnels :

- tunnel dédié à la gestion des clés et des tunnels secondaires
- tunnels destinés à l'échange des données

Le premier tunnel met en place un protocole (IKE) qui utilise généralement les algorithmes à clé publique pour générer un secret partagé. Celui-ci sera ensuite dérivé pour établir les tunnels secondaires (un par AS).

Les tunnels dédiés à l'échange de données utilisent AH pour l'authentification (au niveau machine et non personne humaine) et ESP pour le chiffrement et l'authentification.

Les associations de sécurité (AS) permettent de savoir quelles mesures de sécurité adopter pour un paquet donné (adresses source et destination, port, protocole de transport). C'est un peu similaire à un firewall qui décide, à partir des mêmes critères, de filtrer ou non les paquets. Ici, la décision ne porte pas sur le filtrage mais sur le choix d'une politique de sécurité. D'ailleurs,

on pourrait regrouper ces deux prises de décision en une seule : soit on filtre le paquet, soit on lui applique une politique de sécurité (qui peut être vide).

2.4 Synthèse

Dans cette première partie, nous nous sommes intéressés à différents aspects de la sécurité. Nous avons plus particulièrement insisté sur les aspects relatifs au respect de la vie privée qui constituent le cœur du rapport. Cependant, nous avons également abordé quelques recommandations du w3c qui présentent une façon d'intégrer la sécurité dans des documents XML. Toutes ces recommandations aboutissent finalement à WS-Security qui doit permettre de sécuriser les web services. Nous avons enfin présenté très brièvement les mécanismes de sécurité des couches inférieures pour montrer les similitudes avec ceux de la couche application (WS-Security).

Dans la suite de ce rapport, nous laisserons de côté les aspects de signature et de chiffrement, qui méritaient qu'on s'y intéresse, pour nous focaliser sur la confidentialité à travers P3P.

Chapitre 3

Contributions

3.1 Introduction

Dans cette deuxième partie, nous allons réaliser un évaluateur de politiques P3P en langage ELAN. Cette réalistation nécessite, en premier lieu, la présentation du langage ELAN à travers un exemple qui constitue en lui-même une contribution. Cet exemple consiste à formaliser un firewall (filtrage de paquets) en ELAN. Puis, nous aborderons la partie principale du rapport qui présente la réalisation de l'évaluateur P3P (définition des types, des signatures et implantation des règles de réécriture).

Nous verrons enfin les avantages et les inconvénients d'une telle approche dans le cadre du firewall et de l'évaluateur P3P mais plus globalement de l'utilisation d'ELAN pour formaliser des politiques. Cette analyse sera illustrée par un dernier exemple de modélisation en ELAN qui sera la représentation des domaines introduits par Ponder.

Voyons maintenant comment formaliser le protocole P3P avec ELAN.

3.2 Evaluateur P3P

3.2.1 ELAN

ELAN est un formalisme de spécification qui permet de décrire des signatures multi-sortées, des règles de réécritures conditionnelles et des stratégies de contrôle. Une des particularités d'ELAN est de permettre l'utilisation de grammaires hors contexte pour décrire les signatures. Ceci permet de définir et d'utiliser des opérateurs infixés qui font d'ELAN un langage agréable à utiliser pour spécifier des structures de données complexes telles que celles utilisées dans des prouveurs automatiques, des résolveurs de contraintes ou des outils de modélisation de réseaux par exemple.

La première partie d'une spécification ELAN consiste en la définition des sortes utilisées, la liste des modules importés et un ensemble de règles de grammaires hors contexte pour déclarer les symboles de fonctions. Considérons, par exemple, la signature d'un module permettant de modéliser une politique de firewall. Habituellement, une telle politique de firewall se représente par une table indiquant quels paquets doivent être filtrés et quels paquets doivent être routés. Voici un exemple d'une telle table :

La première ligne du tableau indique qu'un paquet dont le protocole est tcp, l'adresse IP source est 140.192.37.20, le port source et l'adresse IP destination sont quelconques et le port destination est 80, sera filtré.

protocole	IP source	Port source	IP dest	Port dest	decision
tcp	140.192.37.20	any	*.*.*.*	80	deny
tcp	140.192.37.*	any	*.*.*.*	80	accept
tcp	*.*.*.*	any	161.120.33.40	80	accept

TAB. 3.1 – Exemple de règles de firewall

Décrire les règles d'une politique de firewall en ELAN revient à formaliser les informations que l'on va manipuler en donnant une notation. Dire que `tcp` et `udp` sont des protocoles, et dire qu'une adresse IP est constituée de quatre octects séparés par des points, se décrit de la façon suivante en ELAN :

```
tcp     :                           Protocol;
udp     :                           Protocol;
@       : (int)                     Octect;
@.@.@.@ : (Octet Octet Octet Octet) IPAdress;
```

Similairement, nous appelons Header, l'ensemble des informations constituant le membre gauche d'une règle de firewall. Un Header est composé d'un protocole, d'une adresse IP source, d'un port source, d'une adresse IP destination et d'un port destination. En utilisant une notation où chaque élément est séparé par une virgule, cela se traduit de la manière suivante :

```
@         : (int)                                  Port;
@,@,@,@,@ : (Protocol IPAdress Port IPAdress Port) Header;
```

Le formalisme de définition de syntaxe d'ELAN est suffisamment puissant pour décrire tout langage exprimable par des règles de grammaire hors contexte. Cette syntaxe permet de définir et de construire la structure algébrique des données, représentée par des termes.

La deuxième partie d'une spécification ELAN consiste en la définition des opérations manipulant les structures de données précédemment décrites. Le mécanisme d'évaluation élémentaire repose sur la *réécriture* : les règles de réécriture sont des paires de termes (l, r) notées `l => r` et sont utilisées pour définir une relation entre deux termes clos. Nous pouvons ainsi définir un ensemble de règles de réécriture qui rend exécutable une politique de firewall. La traduction de politiques en règles est systématique : il suffit de remplacer ce qui dénotait une valeur quelconque (*any* ou `*`) par une variable ELAN : any, x1,...,x4,y1,...,y4. Les trois règles du tableau 3.1 deviennent ainsi :

```
rules for string
  any                          :int;
  x1,x2,x3,x4,y1,y2,y3,y4      :Octet;
global
  [] eval(tcp,140.192.37.20,any,y1.y2.y3.y4,80) => "deny"   end
  [] eval(tcp,140.192.37.x4,any,y1.y2.y3.y4,80) => "accept" end
  [] eval(tcp,x1.x2.x3.x4,any,161.120.33.40,80) => "accept" end
  ...
end
```

Dans ce dernier exemple, nous avons introduit l'opérateur `eval(@) :(Header) string` qui permet de construire un terme à évaluer. En pratique, pour chaque paquet arrivant, il suffit d'appeler la fonction `eval` pour savoir si le paquet doit être accepté ou refusé. Pour compiler un ensemble de règles, ELAN utilise des techniques de filtrage utilisant des arbres de discrimination.

La complexité de la procédure de filtrage ne dépend donc que de la taille du terme à évaluer et non du nombre de règles du système. Le compilateur ELAN permet ainsi d'appliquer plusieurs millions de règles par seconde, ce qui rend la spécification utilisable en pratique.

3.2.2 Modélisation de P3P avec ELAN

Définition des types

Pour formaliser en ELAN les règles d'un fichier de préférences utilisateur respectant le langage APPEL, nous nous sommes inspirés de la pseudo-grammaire du langage APPEL.

L'ensemble des balises définies dans les langages P3P et APPEL doivent être reconnues comme telles. C'est ce que nous faisons en écrivant (ligne 1 à 3 de la figure suivante) que current, admin, develop, ... sont des noms de balises (element_name). De même, les différents attributs existants sont reconnus grâce aux définitions des lignes 5 à 7. Ensuite, on est capable de reconnaître des entités plus grandes comme les attribute_exp (ligne 9). Par exemple, behavior="request" sera reconnu comme étant de type attribute_exp. Une suite d'attributs appartenant à une même balise est regroupée dans le type attribute_exps : la ligne 10 indique qu'un attribut est un groupe d'attributs contenant un seul élément et la ligne 11 rassemble deux ensembles d'attributs pour créer un autre ensemble qui contient tous leurs attributs.

Notre programme est capable de reconnaître des noms de balises et des attributs. Nous définissons maintenant les balises vides du type `<current/>` ou `<p3p :DATA ref="#user.*"/>`. Une empty-expression est une balise qui a un nom et un certain nombre d'attributs. Voici le formalisme utilisé dans la spécification du langage APPEL :

```
empty-expression=``<''element-name *attribute-expression''/>''
```

Pour traduire cela en ELAN, il suffit d'écrire les lignes 13 et 14 qui sont très proches du formalisme de la grammaire APPEL. Nous réutilisons évidemment les définitions précédentes des types element_name et attribute_exps. De la même façon, les balises qui contiennent d'autres balises sont définies à la ligne 15.

Pour réaliser ces définitions de types, nous nous sommes inspirés du travail rapporté dans l'article [5] sur la réalisation d'un processeur XSLT en ELAN. Reconnaître les composants d'un fichier APPEL ou P3P va nous permettre de le parcourir et de l'évaluer efficacement.

```
current  :ele_name ;                                          1
admin    :ele_name ;                                          2
develop  :ele_name ;                                          3
...                                                           4
ref      :att_name ;                                          5
crtdby   :att_name ;                                          6
behavior :att_name ;                                          7
...                                                           8
@=@      :(att_name string) att_exp ;                         9
@        :(att_exp) att_exps ;                                10
@ @      :(att_exps att_exps) att_exps assocRight ;           11
...                                                           12
<@ @/>   :(ele_name att_exps) empty_exp ;                     13
<@/>     :(ele_name) empty_exp ;                              14
<@>@</@> :(ele_name ctd_exps ele_name) ctg_exp ;              15
...                                                           16
```

Définition des signatures

Nous avons défini cinq fonctions principales qui permettent de faire les manipulations voulues. La fonction appelée lors de la requête initiale est la fonction eval qui prend comme paramètres l'ensemble des règles APPEL et la politique fournie par le site. Elle est responsable de la gestion du parcours des différentes règles (RULE). De plus, elle réalise les appels à la fonction de comparaison cmp. Cette dernière prend comme paramètres deux expressions et les compare. Le résultat attendu est un booléen qui permettra à la fonction eval de savoir si une règle APPEL vérifie la politique. Si c'est la cas, le traitement s'arrête et le résultat est la valeur de l'attribut behavior de la balise RULE.

```
eval(@,@)    :(contained_exps expression) string;
cmp(@,@)     :(contained_exps contained_exps) bool;
cmpEns(@,@,@):(contained_exps contained_exps string) bool;
or(@,@)      :(contained_exps  contained_exps) bool;
orbis(@,@)   :(contained_exps contained_exps) bool;
```

3.2.3 Les règles ELAN

Dans cette section, nous allons voir quelques règles qui donnent une idée générale de l'enchaînement des règles lors de l'évaluation.

```
/****cas generaux******/
[] eval(<RULE behavior=b at>p</RULE>hs,e) => b if cmp(p,e) end
[] eval(<RULE behavior=b at>p</RULE>hs,e) => eval(hs,e)
                                          if not(cmp(p,e)) end

/****cas terminaux*****/
[] eval(<RULE behavior=b at>p</RULE>,e) => b if cmp(p,e) end
[] eval(<RULE behavior=b at>p</RULE>,e) => ""
                                        if not(cmp(p,e)) end
```

La fonction eval prend comme paramètres l'ensemble des règles APPEL et la politique P3P du site web. Dans le cas général, il y a plusieurs règles dans le fichier APPEL. Le résultat de l'évaluation sera le comportement lié à la première règle APPEL rencontrée compatible avec la politique du site. "hs" représente les autres règles APPEL qui suivent la première. "cmp" teste la compatibilité entre une règle APPEL et la politique "e". Dans le cas terminal, il ne reste qu'une règle potentiellement compatible. Si elle l'est, le résultat sera le comportement lié à celle-ci. Sinon, le résultat est une chaîne de caractères vide : ce dernier cas est un cas d'erreur. Normalement, au moins une règle doit s'appliquer pour une politique donnée grâce à la balise OTHERWISE.

On a vu que pour évaluer une règle APPEL, on testait cmp(p,e) où p est la politique incluse dans cette règle et e est la politique du site (evidence). La fonction "cmp" que nous avons implantée n'est pas réservée à POLICY mais peut comparer tout type de balise. Comme on l'a vu dans la section des définitions de types, il existe deux cas principaux : les balises vides `<@/>` et les balises contenantes `<@> @ <@/>`. Dans le premier cas qui consiste à comparer deux balises vides, on a la règle suivante :

```
[] cmp(<a/>,<a/>) => true end
```

Dans le deuxième cas, on a les règles suivantes qui détaillent différents sous-cas : présence d'un connecteur ou pas, d'autres attributs, ...

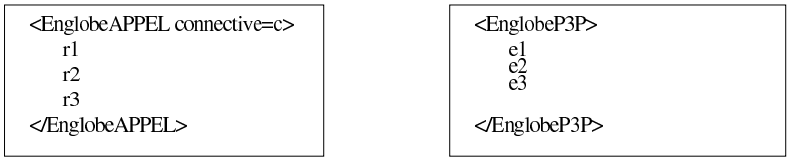

FIG. 3.1 – Comparaison d'ensembles de balises

```
[] cmp(<a appel:connective=co>d</a>,<a>h</a>)
                              => cmpEns(d,h,co)  end
[] cmp(<a appel:connective=co>d</a>,<a at>h</a>)
                              => cmpEns(d,h,co)  end
[] cmp(<a>d</a>,<a>h</a>)
                              => cmpEns(d,h,"and")  end
[] cmp(<a>d</a>,<a at>h</a>)
                              => cmpEns(d,h,"and")  end
```

Il existe un troisième cas intermédiaire qui est la balise DATA. Si la valeur de l'attribut ref a une catégorie variable comme par exemple #dynamic.cookies, il faut comparer la catégorie :

```
[] cmp(<DATA ref=r>d</DATA>,<DATA ref=s>h</DATA>)
      => cmpEns(d,h,"and") and match(r,s)  end
[] cmp(<DATA ref=r/>,<DATA ref=s/>)
      => match(r,s)  end
[] cmp(<DATA ref=r/>,<DATA>d</DATA>)
      => cmpEns(<CATEGORIES>categ(r)</CATEGORIES>,d,"and") end
```

Comparer deux balises englobantes comme à la figure 3.1 revient à comparer leur contenu qui peut être composé de plusieurs balises. Donc on a besoin d'une fonction de comparaison "cmpEns". "cmpEns" a trois paramètres :

- une suite R de balises APPEL
- une suite E de balises P3P
- le connecteur logique qui indique la méthode de comparaison (par défaut "and")

Elle crée récursivement la formule logique à évaluer. Par exemple, si on veut évaluer cmpEns(R,E,"and"), il faut vérifier que R est inclus dans E, ce qui revient à vérifier que :

$$\forall r_i \in R, r_i \in E$$

Pour faire cette vérification, on va construire la formule en appliquant les règles de réécriture cmpEns. La formule précédente équivaut à : [cmp(r1,e1) ou cmp(r1,e2) ou cmp(r1,e3) ou ...] et [cmp(r2,e1) ou cmp(r2,e2) ou cmp(r2,e3) ou ...] et ...

En ELAN :

`[] cmpEns(v ds,hs,"and") => or(v,hs) and cmpEns(ds,hs,"and")` end

On écrit récursivement tous les blocs connectés par " and ". Chaque bloc entre crochets est réalisé par la fonction or :

```
[] or(v,u hs) => cmp(v,u) or or(v,hs) end
```

La procédure d'évaluation comporte un autre aspect qui consiste à comparer les balises DATA et plus particulièrement la valeur de leurs attributs ref. Le rôle particulier de cette balise vient du fait qu'elle contient le type des données collectées par le site. Ces données sont décrites par une notation pointée du type "user.name". Pour comparer deux données respectant cette notation,

nous avons formalisé la structure utilisée et défini une fonction match. Chaque élément (comme user, name ou *) est un token et toute combinaison de tokens est encore un token :

```
* :token;
user :token;
name :token;
bdate :token;
...
@.@ :(token token) token assocRight;
match(@,@) :(token token) bool;
```

Les deux paramètres de la fonction match sont comparés en partant de la gauche. Si l'un des deux est un préfixe de l'autre alors le résultat est "true". Cela se traduit en ELAN par :

```
[] match(x.y,x.z) => match(y,z) end
[] match(x.y,x) => true end
[] match(x,x.z) => true end
[] match(x,x) => true end
```

Le nombre de règles match appliquées pour deux paramètres donnés est au pire égal au nombre de tokens de la plus petite des deux expressions. Par exemple, match ("#dynamic.*", "#dynamic.http.useragent") est évalué en deux règles.

Exemple de réécriture

Voyons maintenant un exemple de réécriture de la fonction cmp :

```
[] cmp(<a appel :connective=co>d</a>,<a>h</a>) => cmpEns(d,h,co) end
```

```
cmp(<RECIPIENT appel:connective="or"><ours/></RECIPIENT>
    ,<RECIPIENT><ours/><same/></RECIPIENT>)
=> cmpEns(<ours/>,<ours/><same/>,"or")
=> or(<ours/>,<ours/><same/>)
=> cmp(<ours/>,<ours/>) or or(<ours/>,<same/>)
=> cmp(<ours/>,<ours/>) or cmp(<ours/>,<same/>)
=> true or false
=> true
```

La comparaison de deux éléments RECIPIENT conduit à la comparaison de leur contenu respectif. On a maintenant comme connecteur "or" et il s'agit de vérifier que {ours} est inclus dans {ours,same}. Cela revient à faire : (ours==ours) ou (ours==same). La première condition est vraie donc le résultat est true. Si p3p :RECIPIENT avait eu des frères, on aurait fait la même chose pour chacun des frères Fi et le résultat aurait été : F1 and F2 and ... and true and ... and Fk où "true" vient de l'exemple que l'on vient d'évaluer et "and" est le connecteur fourni en troisième paramètre de cmp.

Réalisation du prototype d'évaluation

Les balises gérées par l'évaluateur Nous avons fait certains choix pour la réalisation du prototype. En particulier, certaines balises et attributs ne sont pas gérés actuellement comme :

- la balise DISPUTES-GROUP
- la comparaison de <p3p :DATA ref="a"> et <DATA><CATEGORIES>...</CATEGORIES> </DATA> qui nécessite de définir les catégories prédéfinies de P3P concernant les références

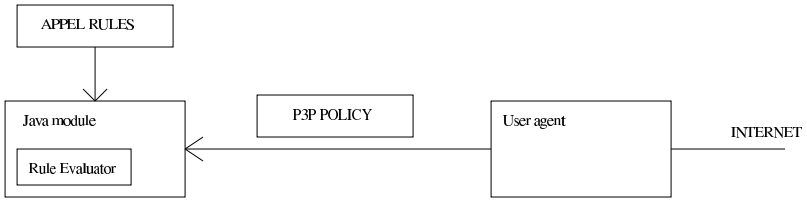

FIG. 3.2 – Module d'évaluation

possibles de DATA (par exemple, la catégorie prédéfinie de user.name est <demographic/>). Nous avons lié certains types de données à leurs caractéristiques pour montrer que c'est possible mais toutes ne sont pas gérées.
- les étoiles (wildcards) dans les attributs ne sont pas gérées.
- les attributs prompt, description, promptmsg de la balise RULE.

Mis à part ces restrictions et certains autres détails peu importants, les balises RULESET, RULE, POLICY, ACCESS, STATEMENT, PURPOSE, RETENTION, RECIPIENT, DATA-GROUP, DATA, CATEGORIES sont supportées par notre prototype.

Intégration dans un navigateur Nous allons intégrer notre prototype dans un navigateur web open-source (Mozilla). Il s'agit de faire un plug-in en C++ ou en java qui récupèrera la politique du site et le fichier de préférences de l'utilisateur. Ces deux fichiers seront transmis à l'évaluateur sous la forme de chaînes de caractères comme à la figure 3.2. La requête suivante :

```
eval(<RULESET> ... </RULESET> , <POLICY> ... <POLICY>) end
```

sera le point de départ de la réécriture.

L'évaluateur sera vu comme un programme externe par le browser et appelé à chaque nouvelle requête http vers un domaine dont la politique n'est pas encore connue ou n'est plus valide.

Choix faits pour l'évaluation L'utilisation d'un outil conçu pour la réécriture à partir de règles est un choix intéressant pour écrire soit directement des politiques, soit des règles qui évaluent des politiques.

En effet, pour la réalisation de notre prototype, nous avons choisi de comparer les règles APPEL et la politique P3P mais nous aurions pu tout à fait transcrire le fichier APPEL en règle ELAN de façon à simplifier son implantation. L'avantage de notre choix est que notre algorithme est utilisable pour tous les fichiers APPEL alors que cette autre possibilité conduirait à faire un fichier ELAN pour les préférences d'un utilisateur unique. Mais elle aurait eu l'avantage d'utiliser encore mieux les possibilités d'ELAN qui est un langage à base de règles. En quelque sorte, on évalue des règles avec d'autres règles.

Cette autre possibilité est d'ailleurs celle que nous avons adoptée pour les politiques de firewall où notre fichier ELAN définit les règles de la politique et non les règles d'évaluation d'une politique décrite ailleurs.

3.3 Évaluation de politiques de sécurité avec ELAN

3.3.1 Modélisation des domaines définis dans Ponder

Considérons un autre exemple de formalisation en langage ELAN. Ponder [10] est un langage qui permet de spécifier des politiques de sécurité et de management pour des systèmes distribués.

Il est développé par l'Imperial College of Science, Technology and Medicine en Grande Bretagne. Ponder définit quatre types de politiques :

- Politique d'autorisation : spécifie quelles opérations un sujet a le droit de faire ou non sur une cible. (politique interprétée par la cible)
- Politique d'obligation : spécifie quelles opérations doit faire un sujet lorsque survient un événement particulier. (politique interprétée par le sujet)
- Politique de refrain : spécifie quelles opérations un sujet n'a pas le droit de faire sur une cible. (politique interprétée par le sujet)
- Politique de délégation : spécifie quelles autorisations un sujet a le droit de déléguer à une cible.

Ponder définit des domaines pour regrouper des entités selon différents critères : situation géographique, type d'entités, responsabilité, ... Une politique s'applique à un groupe et non à une entité unique. De cette façon, lorsqu'une entité disparaît, les politiques liées au groupe restent valables.

Il est facile de définir en ELAN les domaines comme dans le langage Ponder. Par exemple, prenons la structure arborescente de domaines de la figure 3.3.

En ELAN, on peut représenter cela sous la forme :

```
  @ :(DeptManagers) Managers;
//signifie qu'un departement manager est aussi un manager (sorte d'héritage).

  Bob :Employee;
//signifie que Bob est un employé.
```

On reconstruit ainsi l'arbre des domaines en entier très simplement. Pour nommer un domaine, on évite la notation A/B/C mais tous les noms doivent alors être uniques. Une fois les domaines définis, on peut créer des autorisations. Par exemple, si on souhaite que seuls les managers aient accès aux fichiers de salaires, on définit les règles ELAN suivantes :

```
  //définition
  fileAccess(@,@) :(Employees Files) bool;

  //règles
  [] fileAccess(m,pa) => true   end //Un manager a accès aux fichiers de salaires.
  [] fileAccess(e,pa) => false  end //Un employé non manager n'y a pas accès.
  [] fileAccess(e,f)  => true   end //Un employé a accès à tous les autres fichiers.
```

Le premier paramètre est l'équivalent du premier paramètre des autorisations Ponder : subject. Le deuxième paramètre correspond aussi puisqu'il s'agit de la cible : target. Si on fait la requête fileAccess(Bob,FileA) (Bob a-t-il accès à FileA ?), on doit obtenir false car Bob n'a pas le droit d'accès pour ce fichier.

```
enter query term finished by the key word 'end':
fileAccess(Bob,FileA) end

[] start with term :
    fileAccess(Bob,FileA)

       [reduce] start:
       [0]  fileAccess(Bob,FileA)
       [1]  false
```

xpdf: ../docpdf/ponderspec.pdf

Delegation

Consider the following hypothetical domain structure.

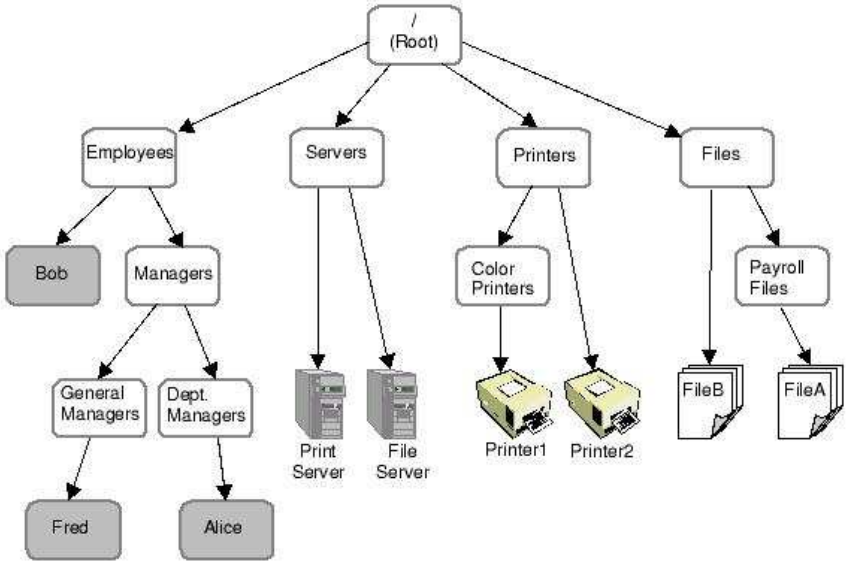

Figure 7. A hypothetical domain

Suppose that the following authorisation policies are in place:

```
type
    auth+ fileAccess (subject S, target files) {
          action read, write;
    } // fileAccess

inst
    auth+ managerFileAccess =
          fileAccess(Employees/Managers, Files/PayrollFiles);

    auth+ employeeFileAccess = fileAccess(/Employees-Employees/Managers,
                       /Files-Files/PayrollFiles);

type
    auth+ printAccess (subject S, target printer) {
          action print;
    } // printAccess

domain man = /Employees/Managers;

inst
    auth+ GMprintAccess =
             printAccess(man/GeneralManagers, Printers/ColorPrinters);

    auth+ employeePrintAccess =
             printAccess(/Employees, /Printers-Printers/ColorPrinters);
```

Ponder Version 2.3 46

Page 46 of 49 +2 Quit

FIG. 3.3 – Exemple de domaine avec Ponder

```
       [reduce] stop :

[] result term:
    false

[] end
```

Si on fait la requête fileAccess(Bob,FileB), on doit obtenir true car Bob a le droit d'accès pour ce fichier.

```
enter query term finished by the key word 'end':
fileAccess(Bob,FileB) end

[] start with term :
    fileAccess(Bob,FileB)

       [reduce] start:
       [0]  fileAccess(Bob,FileB)
       [1]  true
       [reduce] stop :

[] result term:
    true

[] end
```

De la même façon, on peut traduire la politique d'autorisation managerFileAccess de la figure 3.3 par une autre règle ELAN :

```
m :Managers;
pa :PayrollFiles;
...
[] fileAccess(m,pa) => true end
```

Cette règle signifie que les managers (représentés ici par la variable m) sont autorisés à manipuler les fichiers de paies. On peut également écrire en ELAN une politique de délégation. Considérons la politique suivante écrite en langage Ponder :

```
inst
  deleg- invalidDeleg1 (managerFileAccess) {
    subject /Employees/Managers/DeptManagers ;
    grantee /Employees - /Employees/Managers;
  }
```

Cette politique signifie que les DeptManagers (subject) n'ont pas le droit (deleg-) de déléguer la politique d'autorisation décrite précédemment (managerFileAccess) aux employés qui ne sont pas des managers. Elle s'écrit facilement en ELAN :

```
d :DeptManagers;
m :Managers;
e :Employees;
...
[] deleg(d,m) => true end
[] deleg(d,e) => false end
```

La première règle indique qu'un DeptManagers peut déléguer son droit à un manager. La seconde indique qu'un DeptManagers ne peut pas déléguer son droit à un employé. Ici on ne précise pas quelle politique d'autorisation est déléguée mais on pourrait le faire en ajoutant un paramètre. Le but n'est pas de refaire Ponder en ELAN mais de montrer qu'il est possible de faire un équivalent de Ponder simplement en langage ELAN.

Les deux principaux avantages d'ELAN pour cet exemple sont la représentation des domaines et l'écriture des politiques sous forme de règles très simples. Il suffit ensuite d'écrire des requêtes pour savoir si tel employé a accès à tel fichier.

3.3.2 Synthèse

Avantages

Nous avons vu lors de la présentation du langage ELAN qu'il était très simple de décrire une politique de firewall. Pour cela, il suffit de décrire les objets manipulés et d'écrire les règles de façon très naturelle. Cet exemple illustre bien la puissance d'écriture et de modélisation du langage ELAN. On retrouve les mêmes qualités dans la modélisation du langage Ponder qui se prête bien à une traduction vers ELAN.

Inconvénients

Le principal problème lié à l'utilisation d'ELAN dans le cadre des politiques de sécurité est que les règles sont compilées et sont donc statiques. Considérons l'exemple du firewall. Habituellement, les règles qui constituent un firewall doivent être modifiables "à chaud" de façon à ne pas interrompre le service. Si l'on implante notre firewall en ELAN, qu'on le compile en C et qu'on l'intègre au noyau, alors les mises à jour risquent d'être difficiles : recompilation du fichier ELAN, recompilation du noyau. Si on utilise un fichier externe contenant les règles de firewall et qu'on lit ces règles depuis ELAN, on perd les avantages d'ELAN liés à la recherche discriminatoire de la bonne règle.

De façon générale, les politiques de sécurité comme les firewalls, les AS pour IPSEC, ... évoluent dynamiquement et c'est justement là leur force. Donc on peut se demander si les avantages d'ELAN sont suffisants pour faire oublier ses inconvénients.

L'évaluation des politiques P3P réalisée avec ELAN aurait pu être faite avec XSLT. Finalement, notre évaluateur est un parseur qui évalue en plus le contenu (les règles P3P). XSLT est déjà très performant pour parser les fichiers XML donc là aussi, ELAN n'est pas forcément le meilleur outil quand il s'agit de parcourir des documents XML.

Chapitre 4

Conclusion

L'objectif de ce stage était d'acquérir une vision d'ensemble des politiques et des protocoles mettant en place la sécurité sur Internet et plus particulièrement d'étudier les aspects de respect de la vie privée sur Internet. En effet, la sécurité est un domaine vaste qui va de la protection de la vie privée, aux signatures électroniques en passant par les technologies XML. La sécurité est maintenant présente dans plusieurs couches du modèle de l'Internet (applicatif avec WS-Security ou P3P, transport avec SSL, réseau avec IPSec ou encore les firewalls). Cependant, on s'aperçoit rapidement que les mécanismes sur lequels s'appuient tous les protocoles qui traitent de l'intégrité, de l'authentification et de la confidentialité sont les mêmes : fonction de hachage, algorithme symétrique et asymétrique, certificats [12], signatures.

Le travail réalisé s'est principalement focalisé sur la formalisation des règles P3P et leur validation dans un environnement formel basé sur la réécriture. Nous avons notamment réalisé un environnement qui, à partir d'une politique P3P et de préférences utilisateurs en langage APPEL, retourne le comportement à adopter vis à vis d'une ressource sur Internet. Ce prototype utilise le langage ELAN.

L'évaluateur P3P et le firewall, fourni en introduction au langage ELAN, nous ont permis d'avoir un premier aperçu des possibilités offertes par ELAN pour décrire et évaluer des politiques de confidentialité et de sécurité. Nous avons également pu constater les limites de P3P. Par exemple, il ne fournit aucun moyen pour vérifier que la politique annoncée par un site web est bien respectée. C'est pourquoi P3P ne permet pas d'être sûr que les données concernant l'utilisateur ne seront pas utilisées à des fins non attendues. P3P seul n'est pas suffisant mais il peut cependant servir de base dans l'optique d'un Internet où l'utilisateur contrôlerait les informations qu'il souhaite diffuser. Pour compléter P3P, il faudrait un organisme certificateur qui pourrait contrôler les sites implantant P3P. Ainsi, la confiance des utilisateurs serait accrue.

À court terme, des tests de performances et de passage à l'échelle restent à réaliser. En ce qui concerne l'évaluateur P3P, le contrôle permanent des ressources consultées par l'internaute ne doit pas ralentir et donc gêner la navigation. Notre évaluateur a peu d'impact côté serveur donc les tests se feront principalement du côté du client au sein du navigateur Mozilla. Pour le firewall, les attentes en terme de performances sont plus fortes donc nous allons réaliser une plate-forme de tests simulant un traffic important de paquets IP.

Nos perspectives de travaux futurs consistent à étudier plus largement l'utilisation d'ELAN comme un moyen de formaliser des politiques. Le grand avantage d'ELAN est que formaliser les règles revient à réaliser simultanément l'outil qui va les évaluer. Les possibilités d'ELAN pour l'écriture et l'évaluation de politiques doivent maintenant être comparées à celles des langages dédiés à l'expression de politiques orientées sécurité [2]. À plus long terme, il serait intéressant de

pouvoir décrire des politiques de sécurité uniformément, aussi bien pour des protocoles applicatifs que pour des protocoles de niveau réseau. Par exemple, si deux communicants ont certains besoins de sécurité, la plateforme déciderait quels mécanismes mettre en place, après négociation entre les deux entités. Cela existe déjà pour IPSec, mais il faudrait étendre ce mécanisme à tous les niveaux de façon normalisée. Ainsi, on pourrait répondre aux besoins de sécurité dynamiquement. Une autre perspective pour garantir le respect de la vie privée consiste à conserver l'anonymat au niveau réseau par l'utilisation de proxy ou de communauté à routage aléatoire. L'Internaute ne peut donc pas être reconnu par son adresse IP. La difficulté consiste à combiner tous ces mécanismes pour offrir un niveau de sécurité le plus adapté possible aux besoins des applications et des utilisateurs.

Annexe A

Exemple de fichier de références de politique P3P

Ce fichier est le fichier de références de politiques d'IBM. Il signifie que toutes les ressources du site d'IBM appartenant aux chemins /cgi-bin*, /scripts*, ... et tous les cookies du domaine .ibm.com sont couverts par la politique ibm-dynamic. Toutes les ressources qui ne se trouvent pas dans l'un de ces répertoires sont couvertes par la politique ibm-defaut.

```
<META xmlns="http ://www.w3.org/2002/01/P3Pv1">
<POLICY-REFERENCES>
  <!-- Subtrees covered by the 'apps' policy -->
  <POLICY-REF about="http ://www.ibm.com/privacy/p3p/apps.xml#ibm-dynamic">
    <!-- Portions of the site covered -->
    <INCLUDE>/cgi-bin*</INCLUDE>
    <INCLUDE>/scripts*</INCLUDE>
    <INCLUDE>/Scripts*</INCLUDE>
    <INCLUDE>/standards*</INCLUDE>
    <INCLUDE>/Finding*</INCLUDE>
    <INCLUDE>/Find*</INCLUDE>
    <INCLUDE>/Searching*</INCLUDE>
    <INCLUDE>/searching*</INCLUDE>
    <INCLUDE>/Search*</INCLUDE>
    <INCLUDE>/search*</INCLUDE>
    <INCLUDE>/perl*</INCLUDE>
    <INCLUDE>/cgi-perl/*</INCLUDE>
    <INCLUDE>/account/prefs*</INCLUDE>
    <INCLUDE>/gold*</INCLUDE>
    <INCLUDE>/pc/athome*</INCLUDE>
    <INCLUDE>/partnerworld*</INCLUDE>
    <! - Cookies covered ->
    <COOKIE-INCLUDE domain="'.ibm.com'"/>
  </POLICY-REF>
  <! - Subtrees covered by the default policy. ->
  <POLICY-REF about="'http ://www.ibm.com/privacy/p3p/general.xml#ibm-default''>
    <! - Portions of the site covered ->
    <INCLUDE>/*</INCLUDE>
  </POLICY-REF>
</POLICY-REFERENCES>
</META>
```

Annexe B

Exemple de politique P3P

Ce fichier contient la politique ibm-dynamic précédemment citée dans l'annexe A. Au début du fichier, IBM définit un certain nombre de type de données qu'il collecte et qui n'appartiennent pas aux définitions standards (balise DATASCHEMA). Ensuite, on trouve la politique elle-même, introduite par l'élément POLICY.

```
<?xml version="1.0"?>                                                          1
<POLICIES xmlns="http ://www.w3.org/2002/01/P3Pv1">                            2
   <!-- Generated by IBM P3P Policy Editor version Beta 1.11 built 6/4/02 11 :23 AM -->   3
   <!-- Expiry information for this policy -->                                 4
   <EXPIRY max-age="604800"/>                                                  5
                                                                               6
   <!-- Custom data elements defined by this policy. -->                       7
   <DATASCHEMA>                                                                8
   <DATA-DEF name="ibm" short-description="IBM Data">                          9
      <CATEGORIES><uniqueid/></CATEGORIES>                                     10
      <LONG-DESCRIPTION>Custom data elements defined by IBM.</LONG-DESCRIPTION>   11
   </DATA-DEF>                                                                 12
   <DATA-DEF name="ibm.computerinfo" short-description="Computer information">   13
      <CATEGORIES><computer/></CATEGORIES>                                     14
      <LONG-DESCRIPTION>Information about your IBM computer system (for example, a computer   15
          serial number for an IBM personal computer you may have registered online).   16
      </LONG-DESCRIPTION>                                                      17
   </DATA-DEF>                                                                 18
   <DATA-DEF name="ibm.registration" short-description="Registration information">   19
      <CATEGORIES><uniqueid/></CATEGORIES>                                     20
   </DATA-DEF>                                                                 21
   <DATA-DEF name="ibm.registration.userid" short-description="IBM User ID">   22
      <CATEGORIES><uniqueid/></CATEGORIES>                                     23
      <LONG-DESCRIPTION>User ID created by registering for an IBM application or service.   24
      </LONG-DESCRIPTION>                                                      25
   </DATA-DEF>                                                                 26
   <DATA-DEF name="ibm.registration.password" short-description="IBM Password">   27
      <CATEGORIES><uniqueid/></CATEGORIES>                                     28
      <LONG-DESCRIPTION>                                                       29
       Password created by the user when registering for an IBM application or service.   30
      </LONG-DESCRIPTION>                                                      31
   </DATA-DEF>                                                                 32
   <DATA-DEF name="ibm.purchaseinfo" short-description="Purchase information">   33
      <CATEGORIES><preference/><purchase/></CATEGORIES>                        34
      <LONG-DESCRIPTION>Information about products being purchased and payment method.   35
      </LONG-DESCRIPTION>                                                      36
```

```
    </DATA-DEF>
    <DATA-DEF name="ibm.purchaseinfo.payment" short-description="Payment information">
       <CATEGORIES><purchase/></CATEGORIES>
       <LONG-DESCRIPTION>
         Information needed to process payment for an online purchase,
         including credit card type, number, and expiry information.
       </LONG-DESCRIPTION>
    </DATA-DEF>
    <DATA-DEF name="ibm.postings" short-description="Forum posting content">
       <CATEGORIES><content/></CATEGORIES>
       <LONG-DESCRIPTION>
        Content posted to public forums or discussion groups.
       </LONG-DESCRIPTION>
    </DATA-DEF>
    </DATASCHEMA>

<POLICY
    name="ibm-dynamic"
    discuri="http ://www.ibm.com/privacy/"
    opturi="http ://www.ibm.com/privacy/"
    xml :lang="en">
    <!-- Description of the entity making this policy statement. -->
    <ENTITY>
    <DATA-GROUP>
      <DATA ref="#business.name">International Business Machines Corporation</DATA>
      <DATA ref="#business.contact-info.online.email">prvcy@us.ibm.com</DATA>
      <DATA ref="#business.contact-info.online.uri">http ://www.ibm.com/</DATA>
      <DATA ref="#business.contact-info.postal.organization">
      Customer Information Privacy Practices</DATA>
      <DATA ref="#business.contact-info.postal.street">IBM Corporation
       44 S. Broadway</DATA>
      <DATA ref="#business.contact-info.postal.city">White Plains</DATA>
      <DATA ref="#business.contact-info.postal.stateprov">NY</DATA>
      <DATA ref="#business.contact-info.postal.postalcode">10601</DATA>
      <DATA ref="#business.contact-info.postal.country">USA</DATA>
    </DATA-GROUP>
    </ENTITY>
    <!-- Disclosure -->
    <ACCESS><none/></ACCESS>

    <!-- Disputes -->
    <DISPUTES-GROUP>
       <DISPUTES resolution-type="service" service="http ://www.ibm.com/privacy/"
          short-description="Customer Service">
          <LONG-DESCRIPTION>
            Questions regarding this statement should first be directed to IBM. You may
            contact us by e-mail at prvcy@us.ibm.com, or by postal mail at :
            Customer Information Privacy Practices IBM Corporation
            44 S. Broadway White Plains, NY USA 10601
          </LONG-DESCRIPTION>
          <REMEDIES><correct/></REMEDIES>
       </DISPUTES>
       <DISPUTES resolution-type="independent" service="http ://www.truste.org/"
            verification="http ://www.truste.org/validate/331" short-description="TRUSTe">
          <LONG-DESCRIPTION>TRUSTe - building a Web you can believe in.</LONG-DESCRIPTION>
          <IMG src="http ://www.ibm.com/privacy/i/ico_truste.gif" alt="Reviewed by TRUSTe -
             click to verify"/>
          <REMEDIES><correct/></REMEDIES>
       </DISPUTES>
```

```
   </DISPUTES-GROUP>

   <!-- Statement for group ‘‘Universal data’’ -->
   <STATEMENT>
      <EXTENSION optional=‘‘yes’’>
         <GROUP-INFO xmlns=‘‘http ://www.software.ibm.com/P3P/editor/extension-1.0.html’’
          name=‘‘Universal data’’/>
      </EXTENSION>

   <!-- Consequence -->
   <CONSEQUENCE>
This information is collected on all pages within the IBM World Wide Web presence.</CONSEQUENCE>

   <!-- Use (purpose) -->
   <PURPOSE>
    <admin/><contact required=‘‘opt-out’’/><current/><develop/><pseudo-analysis/>
    <pseudo-decision/><individual-analysis required=‘‘opt-out’’/>
    <individual-decision required=‘‘opt-out’’/><tailoring/><telemarketing required=‘‘opt-out’’/>
   </PURPOSE>

   <!-- Recipients -->
   <RECIPIENT><ours/><same required=‘‘opt-out’’/><delivery/></RECIPIENT>

   <!-- Retention -->
   <RETENTION><indefinitely/></RETENTION>

   <!-- Base dataschema elements. -->
   <DATA-GROUP>
   <DATA ref=‘‘#dynamic.clickstream’’/>
   <DATA ref=‘‘#dynamic.http’’/>
   <DATA ref=‘‘#dynamic.clientevents’’/>
   <DATA ref=‘‘#user.home-info.postal.country’’/>
   <DATA ref=‘‘#dynamic.cookies’’><CATEGORIES><computer/><content/><demographic/>
    <interactive/><navigation/><online/><physical/><preference/><purchase/>
    <state/><uniqueid/></CATEGORIES>
   </DATA>
   </DATA-GROUP>
</STATEMENT>

   <!-- Statement for group ‘‘Customer information’’ -->
   <STATEMENT>
      <EXTENSION optional=‘‘yes’’>
         <GROUP-INFO xmlns=‘‘http ://www.software.ibm.com/P3P/editor/extension-1.0.html’’
                  name=‘‘Customer information’’/>
      </EXTENSION>

   <!-- No consequence specified -->
   <!-- Use (purpose) -->
   <PURPOSE><admin/><contact required=‘‘opt-out’’/><current/><develop/><pseudo-analysis/>
    <pseudo-decision/><individual-analysis required=‘‘opt-out’’/>
    <individual-decision required=‘‘opt-out’’/><tailoring/><telemarketing required=‘‘opt-out’’/>
   </PURPOSE>

   <!-- Recipients -->
   <RECIPIENT><ours/><same required=‘‘opt-out’’/><delivery/></RECIPIENT>

   <!-- Retention -->
   <RETENTION><indefinitely/></RETENTION>
```

```
   <!-- Base dataschema elements. -->                                                    155
   <DATA-GROUP>                                                                        156
   <DATA ref="#user.name"/>                                                            157
   <DATA ref="#user.home-info"/>                                                       158
   <DATA ref="#user.business-info"/>                                                   159
   <DATA ref="#user.employer"/>                                                        160
   <DATA ref="#user.department"/>                                                      161
   <DATA ref="#user.jobtitle"/>                                                        162
   <DATA ref="#business.name"/>                                                        163
   <DATA ref="#business.department"/>                                                  164
   <DATA ref="#business.contact-info"/>                                                165
   <DATA ref="#dynamic.miscdata"><CATEGORIES><preference/></CATEGORIES></DATA>         166
   <DATA ref="#dynamic.cookies"><CATEGORIES><computer/><demographic/><online/><physical/>  167
       <preference/><purchase/><state/><uniqueid/></CATEGORIES></DATA>                 168
   </DATA-GROUP>                                                                       169
   <!-- Locally-defined elements. -->                                                  170
   <DATA-GROUP base="">                                                                171
   <DATA ref="#ibm.registration.userid"/>                                              172
   <DATA ref="#ibm.registration.password"/>                                            173
   <DATA ref="#ibm.purchaseinfo.payment"/>                                             174
   <DATA ref="#ibm.computerinfo"/>                                                     175
   </DATA-GROUP>                                                                       176
</STATEMENT>                                                                           177
                                                                                       178
   <!-- Statement for group "Forum Postings" -->                                       179
   <STATEMENT>                                                                         180
      <EXTENSION optional="yes">                                                       181
         <GROUP-INFO xmlns="http ://www.software.ibm.com/P3P/editor/extension-1.0.html"  182
          name="Forum Postings"/>                                                      183
      </EXTENSION>                                                                     184
                                                                                       185
   <!-- Consequence -->                                                               186
   <CONSEQUENCE>                                                                       187
    This group covers any data which is submitted to a public forum,                   188
    such as a public discussion group.                                                 189
   </CONSEQUENCE>                                                                      190
                                                                                       191
   <!-- Use (purpose) -->                                                             192
   <PURPOSE><admin/><current/><develop/></PURPOSE>                                     193
                                                                                       194
   <!-- Recipients -->                                                                195
   <RECIPIENT><ours/><public/></RECIPIENT>                                             196
                                                                                       197
   <!-- Retention -->                                                                 198
   <RETENTION><indefinitely/></RETENTION>                                              199
                                                                                       200
   <!-- Locally-defined elements. -->                                                 201
   <DATA-GROUP base="">                                                                202
   <DATA ref="#ibm.postings"/>                                                         203
   </DATA-GROUP>                                                                       204
</STATEMENT>                                                                           205
                                                                                       206
<!-- End of policy -->                                                                 207
</POLICY>                                                                              208
</POLICIES>                                                                            209
```

Annexe C

Exemple de préférences APPEL

Ce fichier est un fichier de préférences utilisateur en langage APPEL. La première règle du fichier signifie que l'utilisateur doit être averti si des données de type nonident (qui n'identifient pas le visiteur de façon unique) sont collectées dans la ressource courante du site.

```
<appel :RULESET xmlns :appel="http ://www.w3.org/2002/04/APPELv1"
       xmlns :p3p="http ://www.w3.org/2000/12/P3Pv1"
       crtdby="W3C" crtdon="2000-03-15T10 :55 :32+01 :00">
  <appel :RULE behavior="limited" prompt="yes"
          description="Service collects some kind of identifiable
                       information"
          promptmsg="Warning ! Service collects some kind of identifiable
                     information. Do you want to continue (using limited access) ?">
    <p3p :POLICY>
      <p3p :ACCESS appel :connective="non-and">
        <p3p :nonident/>
      </p3p :ACCESS>
    </p3p :POLICY>
  </appel :RULE>
  <appel :RULE behavior="limited" prompt="yes"
          description="Service collects physical and/or online
                       contact information and/or financial account
                       identifiers and/or other data that may be
                       personally-identifiable"
          promptmsg="Warning ! Service collects physical and/or online
                     contact information and/or financial account
                     identifiers and/or other data that may be
                     personally-identifiable. Do you want to
                     continue (using limited access) ?">
    <p3p :POLICY>
      <p3p :STATEMENT>
        <p3p :DATA-GROUP>
          <p3p :DATA>
            <p3p :CATEGORIES appel :connective="or">
              <p3p :physical/>
              <p3p :online/>
              <p3p :uniqueid/>
              <p3p :financial/>
              <p3p :other-category/>
            </p3p :CATEGORIES>
          </p3p :DATA>
        </p3p :DATA-GROUP>
      </p3p :STATEMENT>
```

```
  </p3p :POLICY>                                                          39
 </appel :RULE>                                                           40
 <appel :RULE behavior="request"                                          41
          description="Service does not collect identifiable data or share 42
                       data with other parties">                          43
  <p3p :POLICY>                                                           44
   <p3p :STATEMENT>                                                       45
    <p3p :RECIPIENT appel :connective="and-exact">                        46
     <p3p :ours/>                                                         47
    </p3p :RECIPIENT>                                                     48
    <p3p :PURPOSE appel :connective="non-and">                            49
     <p3p :contact/>                                                      50
     <p3p :telemarketing/>                                                51
     <p3p :individual-analysis/>                                          52
     <p3p :individual-decision/>                                          53
     <p3p :other-purpose/>                                                54
    </p3p :PURPOSE>                                                       55
    <p3p :DATA-GROUP appel :connective="or-exact">                        56
     <p3p :DATA ref="#user.*"/>                                           57
     <p3p :DATA ref="#dynamic.*">                                         58
   <p3p :CATEGORIES><state></p3p :CATEGORIES>                             59
     </p3p :DATA>                                                         60
    </p3p :DATA-GROUP>                                                    61
   </p3p :STATEMENT>                                                      62
  </p3p :POLICY>                                                          63
 </appel :RULE>                                                           64
 <appel :RULE behavior="limited"                                          65
          description="Warning ! Service requests data from your data     66
                       repository or has a practice that doesn't match    67
                       your preferences">                                 68
  <appel :OTHERWISE/>                                                     69
 </appel :RULE>                                                           70
</appel :RULESET>                                                         71
```

Annexe D

Exemple de politique en langage naturel

Ce texte est un extrait de la politique d'IBM en langage naturel qui s'attarde sur l'utilisation des cookies. C'est souvent une politique comme celle-ci qui sert de base pour écrire son équivalent P3P. Le premier paragraphe indique que le site d'IBM collecte des données relatives à l'activité des visiteurs dans le but de faire des statistiques et de savoir quelles sont les pages à la mode. On retrouve l'équivalent de ce paragraphe dans la politique de l'annexe B. Le type de données est dynamic.clickstream, dynamic.http et dynamic.clientevents. Le but de la collecte est tailoring, pseudo-analysis, pseudo-decision, admin comme annoncé dans la politique en langage naturel. Dans le même élément STATEMENT, on retrouve également dynamic.cookies qui permet de réaliser ces statistiques. Ces cookies entrent dans le même contexte d'analyse de comportement utilisateur. C'est donc logique de les retrouver à l'intérieur de la même balise STATEMENT.

```
                                                                    1
We sometimes collect non-identifiable information from              2
visits to our Web sites to help us provide better customer          3
service. For example, we keep track of the domains from             4
which people visit, and we also measure visitor activity on         5
IBM Web sites, but we do so in ways that keep the                   6
information non-identifiable. This information is                   7
sometimes known as "clickstream data." IBM or others on             8
IBM's behalf may use this data to analyze trends and                9
statistics and to help us provide better customer service.          10
                                                                    11
Also, when we collect personal data from you in a                   12
transaction, we may extract some information about that             13
transaction in an non-identifiable format and combine it            14
with other non-identifiable information such as                     15
clickstream data. This information is used and analyzed             16
only at an aggregate level to help us understand trends and         17
patterns. This information is not reviewed at an individual         18
level. If you do not want your transaction details used in          19
this manner you can disable your cookies.                           20
                                                                    21
We collect the information we mentioned in the previous             22
paragraphs through the use of various technologies,                 23
including one called "cookies". A cookie is a piece of data         24
that a Web site can send to your browser, which may then            25
be stored on your computer as an anonymous tag that                 26
```

identifies your computer but not you. Some IBM pages use cookies, sent by IBM or its third party vendors, or other technologies to better serve you when you return to the Web site. You can set your browser to notify you before you receive a cookie, giving you the chance to decide whether to accept it. You can also set your browser to turn off cookies. If you do so, however, some Web sites may not work properly.

Some IBM Web sites also use Web beacon or other technologies to better tailor those sites to provide better customer service. These technologies may be in use on a number of pages across IBM's Web sites. When a visitor accesses these pages, a non—identifiable notice of that visit is generated which may be processed by us or by our suppliers. These Web beacons usually work in conjunction with cookies. If you don't want your cookie information to be associated with your visits to these pages, you can set your browser to turn off cookies.

If you turn off cookies, Web beacon and other technologies will still detect visits to these pages, but the notices they generate cannot be associated with other non-identifiable cookie information and are disregarded. For more information, please see "How to work with Cookies".
We sometimes collect non-identifiable information from visits to our Web sites to help us provide better customer service. For example, we keep track of the domains from which people visit, and we also measure visitor activity on IBM Web sites, but we do so in ways that keep the information non-identifiable. This information is sometimes known as "clickstream data." IBM or others on IBM's behalf may use this data to analyze trends and statistics and to help us provide better customer service.

Also, when we collect personal data from you in a transaction, we may extract some information about that transaction in an non—identifiable format and combine it with other non—identifiable information such as clickstream data. This information is used and analyzed only at an aggregate level to help us understand trends and patterns. This information is not reviewed at an individual level. If you do not want your transaction details used in this manner you can disable your cookies.

We collect the information we mentioned in the previous paragraphs through the use of various technologies, including one called "cookies". A cookie is a piece of data that a Web site can send to your browser, which may then be stored on your computer as an anonymous tag that identifies your computer but not you. Some IBM pages use cookies, sent by IBM or its third party vendors, or other technologies to better serve you when you return to the Web site. You can set your browser to notify you before you receive a cookie, giving you the chance to decide whether to accept it. You can also set your browser to turn off cookies. If you do so, however, some Web sites may not work properly.

Some IBM Web sites also use Web beacon or other technologies to better tailor those sites to provide better customer service. These technologies may be in use on a number of pages across IBM's Web sites. When a visitor accesses these pages, a non-identifiable notice of that visit is generated which may be processed by us or by our suppliers. These Web beacons usually work in conjunction with cookies. If you don't want your cookie information to be associated with your visits to these pages, you can set your browser to turn off cookies.

If you turn off cookies, Web beacon and other technologies will still detect visits to these pages, but the notices they generate cannot be associated with other non—identifiable cookie information and are disregarded. For more information, please see "How to work with Cookies".

Annexe E

Evaluateur P3P/APPEL en ELAN

Cette annexe présente le code de l'évaluateur P3P en ELAN. On trouve au début l'ensemble des mots clés de P3P (balises et attributs), les règles de construction XML (balises ouvrante et fermante) et la définition des règles d'évaluation. Puis, on trouve les règles de réécriture elles-mêmes.

```
module evaluator                                                                    1
  import global                                                                     2
    string bool list[string] eq[element_name] data ;                                3
  end /* for import */                                                              4
                                                                                    5
  sort                                                                              6
    conn connective behav element_name                                              7
    attribute_name expression contained_expression contained_expressions            8
    attribute_expression attribute_expressions containing_expression empty_expression ;  9
  end                                                                               10
                                                                                    11
  operators global                                                                  12
    appel' :'connective=@ :(string) connective ;                                    13
                                                                                    14
    request          :behav ;                                                       15
    limited          :behav ;                                                       16
    block            :behav ;                                                       17
                                                                                    18
    RULESET          :element_name ;                                                19
    RULE             :element_name ;                                                20
    POLICY           :element_name ;                                                21
    ACCESS           :element_name ;                                                22
    STATEMENT        :element_name ;                                                23
    CONSEQUENCE      :element_name ;                                                24
    PURPOSE          :element_name ;                                                25
    RECIPIENT        :element_name ;                                                26
    RETENTION        :element_name ;                                                27
    DATA-GROUP       :element_name ;                                                28
    DATA             :element_name ;                                                29
    CATEGORIES       :element_name ;                                                30
                                                                                    31
    /*****************ACCESS*********************************/                     32
    non-ident        :element_name ;                                                33
    all              :element_name ;                                                34
    contact-and-other  :element_name ;                                              35
    ident-contact    :element_name ;                                                36
    other-ident      :element_name ;                                                37
```

```
none              :element_name ;                                                    38
/*****************PURPOSE*********************************/                         39
current           :element_name ;                                                    40
admin             :element_name ;                                                    41
develop           :element_name ;                                                    42
tailoring         :element_name ;                                                    43
pseudo-analysis   :element_name ;                                                    44
pseudo-decision   :element_name ;                                                    45
individual-analysis  :element_name ;                                                 46
individual-decision  :element_name ;                                                 47
contact           :element_name ;                                                    48
historical        :element_name ;                                                    49
telemarketing     :element_name ;                                                    50
other-purpose     :element_name ;                                                    51
/*****************RECIPIENT*******************************/                         52
ours              :element_name ;                                                    53
delivery          :element_name ;                                                    54
same              :element_name ;                                                    55
other-recipient   :element_name ;                                                    56
unrelated         :element_name ;                                                    57
public            :element_name ;                                                    58
/*****************RETENTION*******************************/                         59
no-retention      :element_name ;                                                    60
stated-purpose    :element_name ;                                                    61
legal-requirement   :element_name ;                                                  62
business-practices   :element_name ;                                                 63
indefinitely      :element_name ;                                                    64
/*****************CATEGORIE*******************************/                         65
physical          :element_name ;                                                    66
online            :element_name ;                                                    67
uniqueid          :element_name ;                                                    68
purchase          :element_name ;                                                    69
financial         :element_name ;                                                    70
computer          :element_name ;                                                    71
navigation        :element_name ;                                                    72
interactive       :element_name ;                                                    73
demographic       :element_name ;                                                    74
content           :element_name ;                                                    75
state             :element_name ;                                                    76
political         :element_name ;                                                    77
health            :element_name ;                                                    78
preference        :element_name ;                                                    79
location          :element_name ;                                                    80
government        :element_name ;                                                    81
other-category    :element_name ;                                                    82
/*********************************************************/                         83
                                                                                     84
p3p' :'@          :(element_name) element_name ;                                     85
appel' :'@        :(element_name) element_name ;                                     86
                                                                                     87
ref               :attribute_name ;                                                  88
crtdby            :attribute_name ;                                                  89
crtdon            :attribute_name ;                                                  90
behavior          :attribute_name ;                                                  91
prompt            :attribute_name ;                                                  92
persona           :attribute_name ;                                                  93
promptmsg         :attribute_name ;                                                  94
service           :attribute_name ;                                                  95
xmlns' :'appel    :attribute_name ;                                                  96
```

```
  xmlns' :'p3p     :attribute_name ;                                                                    97
  //description          :attribute_name ;                                                            98
                                                                                                      99
  @                :(expression) contained_expression ;                                                100
  @=@              :(attribute_name string) attribute_expression ;                                     101
  <@ @ @> @ </@>   :(element_name attribute_expressions connective contained_expressions element_name)  102
                   containing_expression ;                                                            103
  <@ @> @ </@>     :(element_name attribute_expressions contained_expressions element_name)           104
                   containing_expression ;                                                            105
  <@ @> @ </@>     :(element_name connective contained_expressions element_name) containing_expression ; 106
  <@> @ </@>       :(element_name contained_expressions element_name) containing_expression ;          107
  <@ @/>           :(element_name attribute_expressions) empty_expression ;                            108
  <@/>             :(element_name) empty_expression ;                                                  109
  @                :(empty_expression) expression ;                                                    110
  @                :(containing_expression) expression ;                                               111
                                                                                                      112
  @                :(contained_expression) contained_expressions ;                                     113
  @ @              :(contained_expression contained_expressions) contained_expressions ;               114
                                                                                                      115
  @                :(attribute_expression) attribute_expressions ;                                     116
  @ @              :(attribute_expression attribute_expressions) attribute_expressions ;               117
                                                                                                      118
  eval(@,@)        :(contained_expressions expression) string ;                                        119
  cmp(@,@)         :(contained_expressions contained_expressions) bool ;                               120
  cmpEns(@,@,@)    :(contained_expressions contained_expressions string) bool ;                       121
  or(@,@)          :(contained_expressions contained_expressions) bool ;                              122
  orbis(@,@)       :(contained_expressions contained_expressions) bool ;                              123
  egalite(@,@)     :(element_name element_name) bool ;                                                124
  connective(@)    :(contained_expression) string ;                                                   125
  categ(@)         :(string) contained_expressions ;                                                  126
end                                                                                                   127
                                                                                                      128
                                                                                                      129
rules for string                                                                                      130
  a          :element_name ;                                                                          131
  at         :attribute_expressions ;                                                                 132
  co         :connective ;                                                                            133
  r,hs,ds    :contained_expressions ;                                                                 134
  e,p        :expression ;                                                                            135
  b,s        :string ;                                                                                136
global                                                                                                137
                                                                                                      138
  [] eval(<appel :RULESET crtdby="W3C"> r </appel :RULESET> , e ) => eval(r,e) end                    139
  [] eval(<appel :RULE behavior=b> p </appel :RULE>,e) => b if cmp(p,e) end                           140
  [] eval(<appel :RULE behavior=b> p </appel :RULE>hs,e) => b if cmp(p,e) end                         141
  [] eval(<appel :RULE behavior=b> p </appel :RULE>hs,e) => eval(hs,e) end                            142
  [] eval(<appel :RULE behavior=b> p </appel :RULE>,e) => "" end                                      143
                                                                                                      144
  [] connective(<a at appel :connective=s>r</a>) => s end                                             145
  [] connective(<a appel :connective=s>r</a>) => s end                                                146
  [] connective(<a at>r</a>) => "and" end                                                             147
  [] connective(<a>r</a>) => "and" end                                                                148
end                                                                                                   149
                                                                                                      150
                                                                                                      151
rules for bool                                                                                        152
  c,co       :string ;                                                                                153
  enr,ene    :element_name ;                                                                          154
  a,f        :element_name ;                                                                          155
```

```
  d,h,ds,hs  :contained_expressions ;
  v,u,w      :contained_expression ;
  r,s        :string ;
global

  /*****cas terminaux********/
  [] cmp(<a appel :connective=co>d</a>,<f>h</f>) => cmpEns(d,h,co) if egalite(a,f) end
  [] cmp(<a>d</a>,<f>h</f>) => cmpEns(d,h,"and") if egalite(a,f) end
  [] cmp(<a ref=r>d</a>,<f ref=s>h</f>) => cmpEns(d,h,"and")
                              if egalite(a,f) and match(r,s) end
  [] cmp(<a ref=r/>,<f ref=s/>) => egalite(a,f) and match(r,s) end
  [] cmp(<a/>,<f/>)                  => true if egalite(a,f) end
  [] cmp(<a ref=r/>,<f>d</f>) => cmpEns(<p3p :CATEGORIES>categ(r)</p3p :CATEGORIES>,d,"and")
                                       if egalite(a,f) end

  /*****cas generaux********/
  [] cmpEns(v ds,hs,c)   => or(v,hs) and cmpEns(ds,hs,c)        if strcmp(c,"and")==0     end
  [] cmpEns(v ds,hs,c)   => or(v,hs) or cmpEns(ds,hs,c)         if strcmp(c,"or")==0      end
  [] cmpEns(v ds,hs,c)   => not (cmpEns(v ds,hs,"and"))         if strcmp(c,"non-and")==0  end
  [] cmpEns(v ds,hs,c)   => not (cmpEns(v ds,hs,"or"))          if strcmp(c,"non-or")==0  end
  [] cmpEns(ds,v hs,c)   => orbis(ds,v) and cmpEns(ds,hs,c)     if strcmp(c,"or-exact")==0  end
  [] cmpEns(ds,v hs,c)   => cmpEns(ds,v hs,"and")
                            and cmpEns(ds,v hs,"or-exact")      if strcmp(c,"and-exact")==0  end

  [] cmpEns(ds,v,c)  => orbis(ds,v) if strcmp(c,"or-exact")==0  end
  [] cmpEns(v,hs,c)  => or(v,hs)                          end

  [] or(v,u hs)   => cmp(v,u) or or(v,hs)   end
  [] or(v,u)      => cmp(v,u)               end
  [] orbis(u ds,v) => cmp(u,v) or orbis(ds,v)  end
  [] orbis(u,v)   => cmp(u,v)               end

  /****Egalite de deux noms de balises****/
  [] egalite(p3p :enr,ene) => enr==ene end
  [] egalite(appel :enr,ene) => enr==ene end
  [] egalite(enr,ene) => enr==ene end
end


rules for contained_expressions
global
  [] categ("#user.name")       => <physical/><demographic/>  end
  [] categ("#user.bdate")      => <demographic/>             end
  [] categ("#user.login")      => <uniqueid/>                end
  [] categ("#user.cert")       => <uniqueid/>                end
  [] categ("#user.gender")     => <demographic/>             end
  [] categ("#user.employer")   => <demographic/>             end
  [] categ("#user.department")  => <demographic/>            end
  [] categ("#user.jobtitle")   => <demographic/>             end
  [] categ("#user.home-info")  => <physical/><online/><demographic/>  end

  [] categ("#business.name")   => <demographic/>             end
  [] categ("#business.department")  => <demographic/>        end
  [] categ("#business.cert")   => <uniqueid/>                end
  [] categ("#business.contact-info")  => <physical/><online/><demographic/>  end

  [] categ("#dynamic.clickstream")  => <navigation/><computer/>  end
  [] categ("#dynamic.http")    => <navigation/><computer/>   end
  [] categ("#dynamic.clientevents")  => <navigation/>        end
```

```
   [] categ("#dynamic.searchtext")  => <interactive/>          end
   [] categ("#dynamic.interactionrecord")=> <interactive/>     end

  end
end
```

Annexe F

Modélisation de Ponder en ELAN

Cette annexe présente une modélisation restreinte des domaines au sens de Ponder et quelques règles définissant des droits d'accès. Par exemple, à la ligne 52, on apprend que les GeneralManagers ont le droit d'imprimer en couleur.

```
module ponder                                                                    1
  import global                                                                  2
    string bool list[string] ;                                                   3
  end /* for import */                                                          4
                                                                                 5
  sort                                                                           6
    Root Employees Servers Printers Files Managers                               7
    ColorPrinters PayrollFiles GeneralManagers DeptManagers ;                    8
  end                                                                            9
                                                                                 10
  operators global                                                               11
    @                 :(DeptManagers) Managers ;                                 12
    @                 :(GeneralManagers) Managers ;                              13
    @                 :(Managers) Employees ;                                    14
    @                 :(Employees) Root ;                                        15
    @                 :(Servers) Root ;                                          16
    @                 :(ColorPrinters) Printers ;                                17
    @                 :(Printers) Root ;                                         18
    @                 :(PayrollFiles) Files ;                                    19
    @                 :(Files) Root ;                                            20
    Bob               :Employees ;                                               21
    Fred              :GeneralManagers ;                                         22
    Alice             :DeptManagers ;                                            23
    PrintServer       :Servers ;                                                 24
    FileServer        :Servers ;                                                 25
    Printer1          :ColorPrinters ;                                           26
    Printer2          :Printers ;                                                27
    FileB             :Files ;                                                   28
    FileA             :PayrollFiles ;                                            29
                                                                                 30
    printAccess(@,@)  :(Employees Printers) bool ;                               31
    fileAccess(@,@)   :(Employees Files) bool ;                                  32
    deleg(@,@)        :(Employees Employees) bool ;                              33
    write(@,@)        :(Employees Files) bool ;                                  34
    read(@,@)         :(Employees Files) bool ;                                  35
  end                                                                            36
                                                                                 37
rules for bool                                                                   38
```

```
  g   :GeneralManagers ;                                                    39
  d   :DeptManagers ;                                                       40
  m   :Managers ;                                                           41
  c   :ColorPrinters ;                                                      42
  p   :Printers ;                                                           43
  e   :Employees ;                                                          44
  pa  :PayrollFiles ;                                                       45
  f   :Files ;                                                              46
 global                                                                     47
                                                                            48
  [] read(e,f)    => fileAccess(e,f) end                                    49
  [] write(e,f)   => fileAccess(e,f) end                                    50
                                                                            51
  [] printAccess(g,c) => true end                                           52
  [] printAccess(e,c) => false end                                          53
  [] printAccess(e,p) => true end                                           54
                                                                            55
  [] fileAccess(m,pa) => true end                                           56
  [] fileAccess(e,pa) => false end                                          57
  [] fileAccess(e,f) => true end                                            58
                                                                            59
  [] deleg(d,m) => true end                                                 60
  [] deleg(d,e) => false end                                                61
  [] deleg(g,m) => true end                                                 62
  [] deleg(g,e) => false end                                                63
 end                                                                        64
                                                                            65
end                                                                         66
```

Annexe G

Glossaire

APPEL : Langage défini par le W3C permettant à un internaute de décrire ses préférences en matière de traîtement des données personnelles

ELAN : Langage à base de règles fondé sur le principe de réécriture

HTTP : Protocole applicatif utilisé pour transporter des fichiers (principalement HTML) sur Internet

IPSec : Protocole de niveau 3 permettant de créer des tunnels sécurisés

PONDER : Langage pour spécifier des politiques de management et de sécurité pour les systèmes distribués

SSL : Protocole de niveau 4 permettant le chiffrement des échanges sensibles et le contrôle d'accès à certaines applications

URI : Uniform Ressource Identifiers

W3C : World Wide Web Consortium, organisme publiant des recommandations principalement basées sur XML

WS-Security : Protocole permettant d'intégrer la sécurité dans les web services

X-KISS : Protocole pour sous-traîter la gestion des clés

X-KRSS : Protocole pour permettre l'inscription à un service de gestion de clés

X509 : Certificat attribué à un unique utilisateur contenant des informations telles que le nom de l'utilisateur et la clé publique. Le certificat est signé avec la clé privée de l'autorité de certification

XKMS : Protocol regroupant X-KISS et X-KRSS

XML : Langage de balises défini par le W3C

Bibliographie

[1] B. Atkinson al. Web services security (ws-security). *http ://www.ibm.com/developerworks/library/ws-secure/*, 2002.

[2] S. Duflos, G. Diaz, V., and E. Horlait. A comparative study of policy specification languages for secure distributed applications. *DSOM 2002,LNCS 2506*, 1 :157–168, 2002.

[3] H. Hamed E. S. Al-Shaer. Firewall policy advisor for anomaly discovery and rule editing. *IM 2003*, pages 17–30, 2003.

[4] H. Hochheiser. The platform for privacy preference as a social protocol : An examination within the u.s. policy context. *ACM Transactions on Internet Technology (TOIT)*, 2(4) :276–306, 2002.

[5] C. Kirchner, Z. Qian, P. K. Singh, and J. Stuber. Xemantics : a rewriting calculus-based semantics of xslt. Technical Report A01-R-386, INRIA, 2002.

[6] D. M. Kristol. Http cookies : Standards, privacy, and politics. *ACM Transactions on Internet Technology (TOIT)*, 1(2) :151–198, 2001.

[7] M. Marchiori L. Cranor, M. Langheinrich. A p3p preference exchange language 1.0. Draft, W3C, 2002.

[8] M. Marchiori M. Presler-Marshall J. Reagle L. Cranor, M. Langheinrich. The platform for privacy preferences 1.0 (p3p1.0) specification. Specification, W3C, 2002.

[9] B. Fox B. LaMacchia-E. Simon M. Bartel, J. Boyer. Xml-signature syntax and processing. Recommendation, W3C, 2002.

[10] E. Lupu M. Sloman N. Damianou, N. Dulay. Ponder : A language for specifying security and management policies for distributed systems. *Imperial College Research Report DoC*, 2000.

[11] H. Dubois C. Kirchner-H. Kirchner P.-E. Moreau Q.-H. Nguyen C. Ringeissen M. Vittek P. Borovansky, H. Cirstea. Elan user manual. manuel utilisateur, INRIA, 2002.

[12] P. Leca S. Aumont, C. Gross. Certificats x509 et infrastructure de gestion de clés. 2001.

[13] C. Vidal. Tutoriel soap. Tutoriel, planeteXML.com.

[14] B. Fox B. Dillaway Brian LaMacchia-J. Epstein J. Lapp W. Ford, P. Hallam-Baker. Xml key management specification (xkms). Note, W3C, 2001.